Polynomiaikainen alkulukutestaus

Jade Saareke

6. syyskuuta 2018

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI
MATEMAATTIS-LUONNONTIETEELLINEN TIEDEKUNTA
MATEMATISK-NATURVETENSKAPLIGA FAKULTETEN
FACULTY OF SCIENCE

Tiedekunta – Fakultet – Faculty	Koulutusohjelma – Utbildningsprogram – Degree programme
Matemaattis-luonnontieteellinen	Matematiikan aineenopettaja

Tekijä – Författare – Author
Jade Saareke

Työn nimi – Arbetets titel – Title
Polynomiaikainen alkulukutestaus

Työn laji – Arbetets art – Level	Aika – Datum – Month and year	Sivumäärä – Sidoantal – Number of pages
Pro gradu -tutkielma	Syyskuu 2018	42

Tiivistelmä – Referat – Abstract

Tutkielmassa perehdytään vaativuusteorian näkökulmasta lukuteoreettiseen ongelmaan alkuluvun tunnistamisesta.Työn aiheena on alkulukutestauksen polynomiaikaisuus ja keskiössä on tämän osoittava matemaattinen tulos, ehdoton deterministinen polynomisessa ajassa toimiva alkulukutestausalgoritmi. Tutkielmassa käydään läpi kyseisen algoritmin oikean toiminnan ja aikavaativuuden todistukset.

Alkulukutestauksessa pyritään selvittämään, onko annettu luku alkuluku. Vaativuusteoriassa tämä kysymys muotoillaan kielen PRIMES päätösongelmaksi. Päätösongelman ratkaisuun haetaan algoritmia, joka kertoo vastauksen kysymykseen, kuuluuko annettu syöte joukkoon vai ei. Algoritmin katsotaan olevan tehokas, jos se on vaativuusluokassa P. Vaativuusluokka P on deterministisen polynomiaikaisten algoritmien luokka.

Tutkielman ensimmäinen luku on johdanto aiheeseen. Toisessa luvussa käsitellään vaativuusteoriaa. Käydään läpi aikavaativuuden analysointia asymptoottisen analyysin keinoin ja tutustutaan Turingin koneisiin sekä niiden algoritmiyhteyteen. Lopuksi esitellään vaativuusluokat P, NP ja co-NP.

Kolmas luku sisältää lukuteorian perusteita ja erityisesti alkulukuja koskevia tuloksia. Aluksi käsitellään jaollisuutta ja kongruenssia sekä binomikertoimia, minkä jälkeen keskitytään alkulukuihin. Perehdytään alkuluvun tunnistamiseen ja lopuksi tuodaan mukaan vaativuusteorian näkökulma alkuluvuntunnistamisongelman PRIMES esittelyn muodossa.

Neljännessä luvussa käsitellään algebran sisältöjä. Keskeisinä aiheina ovat äärelliset kunnat ja syklotomiset polynomit. Tutkielman viidennessä luvussa käydään läpi todistus alkulukutestauksen polynomiaikaisuudelle. Käsitellään tarkoitukseen kehitetyn ehdottoman deterministisen algoritmin toiminta, sen todistus ja polynomiaikainen aikavaativuus.

Avainsanat – Nyckelord – Keywords

Säilytyspaikka – Förvaringställe – Where deposited
Kumpulan tiedekirjasto

Muita tietoja – Övriga uppgifter – Additional information

Sisältö

1 Johdanto **4**

2 Laskennan vaativuuden analysointi **6**
2.1 Aikavaativuus 6
2.2 Turingin kone ja algoritmit 8
2.3 Vaativuusluokat P, NP ja co-NP 10

3 Alkuluvut **12**
3.1 Jaollisuus ja kongruenssi 12
3.2 Binomikertoimet 13
3.3 Alkuluvut 16
3.4 Alkuluvun tunnistaminen 18
3.5 PRIMES 19

4 Äärelliset kunnat ja syklotomiset polynomit **20**
4.1 Ryhmät, renkaat ja kunnat 20
4.2 Syklotomiset polynomit 23

5 Polynomiaikainen algoritmi alkulukutestaukseen **25**
5.1 Algoritmi 25
5.2 Algoritmin oikeellisuus 28
5.3 Introspektiivisuus 31
5.4 Algoritmin oikeellisuuden todistaminen 33
5.5 Algoritmin aikavaativuus 38

Luku 1

Johdanto

Tässä tutkielmassa perehdytään alkulukujen tunnistamiseen polynomisessa ajassa. Tutustutaan tarkoitukseen kehitettyyn algoritmiin, jonka Agrawal, Kayal ja Saxena ovat kehittäneet ja esittelevät artikkelissaan PRIMES is in P [1]. Kyseessä on ehdoton deterministinen polynomisessa ajassa toimiva algoritmi, joka osaa sanoa syötteenä saamastaan luvusta, onko se alkuluku vai ei.

Polynomiaikaisessa alkulukutestauksessa yhdistyvät vaativuusteoria ja lukuteoria, joiden lisäksi tarvitaan joitakin työkaluja algebrasta. Vaativuusteoria käsittelee laskennan tehokkuutta ja alkuluvut kuuluvat lukuteorian aihepiireihin. Tutkielman lähteinä on käytetty aiheeseen liittyviä kirjoja sekä muutamaa artikkelia.

Agrawalin, Kayalin ja Saxenan algoritmi kuuluu vaativuusluokkaan P, mikä tarkoittaa, että se on deterministisesti polynomiaikainen. Deterministisellä algoritmilla on vain yksi mahdollinen suorituspolku jokaiselle syötteelle. Algoritmi siis toimii kaikissa tilanteissa polynomiaikaisesti, eikä se valitse tai arvaa erilaisten laskentavaihtoehtojen välillä.

Algoritmi on erityinen siksi, että sen löytyminen osoitti, että ongelma siitä, onko luku alkuluku, kuuluu luokkaan P. Ennen tämän algoritmin keksimistä ei ollut ehdotonta deterministisesti polynomiaikaista algoritmia alkulukutestauksen tarkoitukseen. Aiemmat algoritmit alkulukujen tunnistamiseen ovat olleet deterministisiä vain jonkin oletetun lisäehdon alaisuudessa, toimineet polynomisessa ajassa, mutta tuottaneet oikean tuloksen vain jollakin todennäköisyydellä, olleet epädeterministisiä tai niiden suoritusaika ei ole ollut missään määrin polynominen.

Tässä tutkielmassa on viisi lukua, joista ensimmäinen on tämä johdanto. Toisessa luvussa käsitellään laskennan vaativuutta. Tutustutaan siihen, mitä oikeastaan tarkoittaa polynomiaikainen algoritmi. Kolmannessa luvussa aiheena ovat alkuluvut. Perehdytään alkulukujen ominaisuuksiin ja alkuluvun

tunnistamiseen. Esitellään vaativuusteoreettinen alkulukujentunnistamisongelma PRIMES.

Neljännessä luvusssa käsitellään algebran sisältöjä, jotka ovat oleellisia viidennen luvun todistuksissa. Näihin kuuluvat muun muassa äärelliset kunnat ja syklotomiset polynomit. Viidennessä luvussa tutustutaan artikkelissa [1] esiteltyyn algoritmiin ja käydään läpi siihen liittyvät oikeellisuustodistus ja aikavaativuusanalyysi.

Luku 2

Laskennan vaativuuden analysointi

Tässä luvussa käsitellään tutkielman aiheen kannalta oleellista vaativuusteorian sisältöä. Osiossa 2.1 tutustutaan algoritmien aikavaativuuteen ja asymptoottiseen analyysiin. Osiossa 2.2 puolestaan keskitytään Turingin koneeseen ja osiossa 2.3 perehdytään vaativuusluokkiin P, NP ja co-NP.

2.1 Aikavaativuus

Algoritmin suoritusaikaan vaikuttaa olennaisesti algoritmin laskennallinen tehokkuus, *aikavaativuus*, joka kuvaa algoritmin käyttämää laskenta-aikaa. Tämä ilmaistaan sen suorittamien *perusoperaatioiden*, eli algoritmin suorittamien yksinkertaisimpien laskenta-askelien, määränä. Yksinumeroisten lukujen yhteen- ja kertolasku ovat perusoperaatioita.

Suuren määrän operaatioita suorittaminen vie enemmän aikaa kuin muutaman operaation suorittaminen, joten algoritmin suoritusaika riippuu suoraan sen suorittamien perusoperaatioiden määrästä. Aikavaativuus esitetään tyypillisesti algoritmin saaman syötteen pituuden funktiona.

Algoritmien suorittamien perusoperaatioiden määrät ovat usein monimutkaisia ilmaista, joten niitä tyydytään yleensä vain arvioimaan. Tämän voi tehdä arvioimalla aikavaativuutta perusoperaatioiden määrää ilmaisevan funktion nopeiten kasvavan termin avulla.

Suurilla syötteillä nopeiten kasvava termi dominoi suoritettavien perusoperaatioiden määrää. Tästä johtuen kelvollinen arvio saadaan, vaikka hitaammin kasvavat termit ja termien kertoimet jätetään huomiotta. Näin tehdään asymptoottisessa analyysissä, jossa seuraavaksi määriteltävät merkinnät ovat käytössä.

Esitellään aluksi O-merkintä, jonka avulla voidaan tarkastella suoritusajan ylärajaa.

Määritelmä 2.1. Olkoot f ja g funktioita luonnollisilta luvuilta epänegatiivisille reaaliluvuille. Merkitään $f(n) = O(g(n))$, jos on olemassa positiiviset kokonaisluvut c ja n_0, joilla $f(n) \leq cg(n)$ kaikilla kokonaisluvuilla $n \geq n_0$.

Kun $f(n) = O(g(n))$, sanotaan, että $g(n)$ on asymptoottinen yläraja funktiolle $f(n)$. Seuraavassa esimerkissä havainnollistetaan funktion ylärajan arviointia O-merkintää käyttäen.

Esimerkki 2.2. Olkoon $f(n) = 5n^3 + 7n^2 + 2n + 3$. Tällöin $f(n) = O(n^3)$, sillä voidaan valita $c = 6$ ja $n_0 = 10$, jolloin $5n^3 + 7n^2 + 2n + 3 \leq 6n^3$ kaikilla $n \geq 10$. Vastaavasti pätee myös $f(n) = O(n^4)$, sillä $n^4 > n^3$.

Seuraavaksi määritellään muita merkintöjä aikavaativuuden arviointiin.

Määritelmä 2.3. Olkoot f ja g funktioita luonnollisilta luvuilta epänegatiivisille reaaliluvuille.

1. $f(n) = \Omega(g(n))$, jos $g(n) = O(f(n))$.
2. $f(n) = \Theta(g(n))$, jos $f(n) = O(g(n))$ ja $g(n) = O(f(n))$.
3. $f(n) = o(g(n))$, jos kaikilla $\epsilon > 0$ riittävän suurilla n pätee

$$f(n) \leq \epsilon \cdot g(n).$$

4. $f(n) = \omega(g(n))$, jos $g(n) = o(f(n))$.

Merkintä Ω sopii siis alarajan tarkasteluun ja merkintää $f(n) = \Theta(g(n))$ käytetään, jos $f(n) = O(g(n))$ ja $f(n) = \Omega(g(n))$. Merkinnät o ja ω vastaavat muuten merkintöjä O ja Ω, mutta eivät salli asymptoottista yhtäsuuruutta.

Määritellään vielä $O^{\sim}$-merkintä.

Määritelmä 2.4. Funktiolle $f : \mathbb{N} \to \mathbb{R}^+$, jonka raja-arvo $\lim_{n\to\infty} f(n) = \infty$, pätee

$$O^{\sim}(f) = \left\{ g | g : \mathbb{N} \to \mathbb{R}^+, \exists C > 0 \exists n_0 \exists k \forall n \geq n_0 : g(n) \leq C \cdot f(n) \log(f(n))^k \right\}.$$

Määrittely voitaisiin tehdä myös käyttämällä O- ja poly-merkintöjä. Merkintä $f(n) = \text{poly}(n)$ tarkoittaa, että $f(n)$ on n:n jonkin polynomin ylhäältä rajoittama. Funktiolle $f(n)$ merkintä $O^{\sim}(f(n))$ tarkoittaa samaa kuin $O(f(n) \cdot \text{poly}(\log f(n)))$.

$O^{\sim}$-merkintä on hyödyllinen merkintöjen lyhentämisessä ja sen avulla huomio voidaan keskittää dominoivan termin merkitsevimpään tekijään. Esimerkiksi aikavaativuus $O((\log n)(\log\log n)(\log\log\log n))$ voidaan esittää muodossa $O^{\sim}(\log n)$.

Luvun n binääriesityksen pituus, ja siten syöteluvun n koko, on $\lceil \log n \rceil$. Aikavaativuus lasketaan syötteen koon mukaan. Kahden binääriesitykseltään korkeintaan m-pituisen luvun summa, tulo ja osamäärä voidaan suorittaa ajassa $O^{\sim}(m)$ (todistukset katso esimerkiksi [7]).

2.2 Turingin kone ja algoritmit

Turingin kone on Alan Turingin vuonna 1936 esittelemä malli. Turingin koneella voidaan mallintaa laskentaa.

Turingin koneen muisti on rajoittamaton. Tämä ilmenee koneessa ääretönpaikkaisena *nauhana*, jolta kone voi lukea ja jolle se voi kirjoittaa symboleita. Kone voi liikkua nauhalla ja siten lukea ja kirjoittaa eri kohtia.

Turingin koneella on *tiloja*, joissa se voi olla. Kone lähtee liikkeelle *alkutilasta*. Tilasiirtymiin näiden tilojen välillä liittyy myös nauhalla liikkumista, lukemista ja kirjoittamista. Kone ei voi siirtyä tilojen välillä mielivaltaisesti, vaan mahdolliset tilasiirtymät ja niihin liittyvät nauhaoperaatiot on koneessa ennalta määritelty *siirtymäfunktiona*.

Turingin koneen laskenta päättyy, kun kone siirtyy ennalta määrättyyn *hyväksyvään tilaan* tai *hylkäävään tilaan*. Esimerkiksi, jos kyseessä oleva Turingin kone selvittää, kuuluuko sille syöttenä annettu luku tiettyyn joukkoon, ja kone päätyy hylkäävään tilaan, syötteenä saatu luku ei kuulu joukkoon. Vastaavasti tässä tapauksessa hyväksyvään tilaan päätyminen kertoo luvun kuulumisesta joukkoon.

Seuraavana on formaali määritelmä Turingin koneelle.

Määritelmä 2.5. *Turingin kone* on seitsikko $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$, jossa Q on äärellinen joukko tiloja, Σ äärellinen symbolia $\textvisiblespace$ sisältämätön syöteaakkosto, Γ äärellinen symbolin $\textvisiblespace$ ja syöteaakkoston sisältävä nauhaaakkosto, $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ siirtymäfunktio, $q_0 \in Q$ alkutila, $q_{accept} \in Q$ hyväksyvä tila ja $q_{reject} \in Q$ alkutilasta eroava hylkäävä tila.

Määritelmän Turingin kone on *deterministinen* ja *yksinauhainen*. Turingin kone voi kuitenkin olla myös *epädeterministinen* kuten myös *moninauhainen*. Seuraavaksi kerrotaan lyhyesti deterministisen ja epädeterministisen sekä yksinauhaisen ja moninauhaisen Turingin koneen eroista ja yhtäläisyyksistä.

Turingin kone voi olla deterministinen tai epädeterministinen. Deterministinen Turingin kone ei joudu missään vaiheessa arvaamaan tai valitsemaan, minkä siirtymän se tekee. Samasta tilasta ei voi samalla syötemerkillä siirtyä usealla eri tavalla, vaan kaikki siirtymät on määritelty yksikäsitteisesti.

Koneen laskenta voi edetä samalla syötteellä vain yhdellä tavalla. Epädeterministinen Turingin kone puolestaan voi kokeilla useita vaihtoehtoja. Tilasiirtymäfunktiossa samasta tilasta voi samalla merkillä edetä useammalla kuin yhdellä tavalla. Jos jokin laskennan vaihtoehto johtaa hyväksyvään tilaan, epädeterministinen Turingin kone hyväksyy.

Turingin kone voi olla yksinauhainen tai moninauhainen. Moninauhaisessa Turingin koneessa on nimensä mukaan useampi kuin yksi nauha ja muuten kone on samanlainen kuin tavallinen Turingin kone. Näillä nauhoilla on omat nauhapäät ja usein myös eriytyneet käyttötarkoitukset. Esimerkiksi syöte voi olla yhdellä nauhalla ja toinen nauha on työnauha.

Kaikille epädeterministisille ja moninauhaisille Turingin koneille on vastaavat deterministiset ja yksinauhaiset Turingin koneet. Vaikka koneet pystyvät simuloimaan toisiaan, niiden tähän laskentaan kuluttama aika ei aina ole sama.

Turingin konetta voi käyttää funktion laskemiseen. Turingin kone *laskee funktion* siten, että se määrittää saamansa syötteen perusteella funktion arvon kyseisellä syötteellä ja kun kone pysähtyy, määritetty funktion arvo on nauhalla.

Totuusarvofunktio saa arvonsa joukosta $\{0,1\}$. Totuusarvofunktioon f liittyvää joukkoa $L_f = \{x | f(x) = 1\} \subseteq \{0,1\}^*$ kutsutaan *kieleksi*. Kieli on siis merkeistä 0 ja 1 koostuvien merkkijonojen joukko, jonka alkioilla on jokin yhteinen totuusarvofunktion määräämä ominaisuus.

Laskennallinen ongelma funktion f laskemisesta voidaan samastaa kielen L_f *päätösongelman* kanssa. Päätösongelmassa kysytään, kuuluuko annettu merkkijono kieleen. Kun koneelle on syötteenä annettu kyseinen merkkijono, lasketun totuusarvofunktion arvo 1 tarkoittaa, että merkkijono kuuluu kieleen, kun taas arvo 0 kertoo kieleen kuulumattomuudesta.

Määritellään seuraavaksi kaksi ominaisuutta, jotka voivat kuvata kieltä. Aloitetaan tunnistettavuudella. Turingin kone *tunnistaa* kielen, jos se hyväksyy kaikki kieleen kuuluvat merkkijonot, mutta ei muita merkkijonoja.

Määritelmä 2.6. Kieli on *Turing-tunnistettava*, jos jokin Turingin kone tunnistaa sen.

Tunnistettavuus ei takaa, että olisi olemassa Turingin kone, joka sekä hyväksyisi kaikki kieleen kuuluvat merkkijonot että hylkäisi kaikki kyseiseen kieleen kuulumattomat merkkijonot. Turingin kone voi paitsi hyväksyä tai hylätä syötteen, myös jäädä silmukkaan. Silmukkaan jäädessään kone ei pysähdy koskaan, vaan jatkaa kiertoaan samoissa tiloissa loputtomiin.

On olemassa Turingin koneita, jotka eivät koskaan päädy silmukkaan, vaan pysähtyvät kaikilla syötteillä. Näitä Turingin koneita kutsutaan *ratkaisijoiksi*, koska ne aina ratkaisevat hyväksyvätkö vai hylkäävätkö syötteen.

Kun ratkaisija tunnistaa kielen, sanotaan, että se *ratkaisee* kyseisen kielen. Totuusarvofunktion kannalta tämä tarkoittaa sitä, että kone laskee funktion ja että vain kaikki kieleen kuuluvat syötteet tuottavat funktion arvoksi 1.

Määritelmä 2.7. Kieli on *Turing-ratkeava* tai lyhemmin *ratkeava*, jos jokin Turingin kone ratkaisee sen.

Turingin koneisiin liittyvät myös algoritmit. Algoritmi on kokoelma ohjeita jonkin tehtävän suorittamiseksi. Tämä tiivistää epämuodollisesti algoritmin ajatuksen. Varsinainen määritelmä algoritmille saatiin vuonna 1936, kun Alonzo Churchin ja Alan Turingin määritelmät algoritmille osoittautuivat yhtäpitäviksi. Tästä muodostui algoritmikäsitteen ja tarkan määritelmän yhdistävä *Churchin-Turingin teesi*.

Churchin-Turingin teesi sanoo, että Turingin koneella voidaan simuloida mitä tahansa fyysisesti toteutettavissa olevaa laskentalaitetta. Churchin-Turingin teesin mukaan siis näiden fyysisesti toteutettavissa olevien laskentalaitteiden algoritmit ovat toteutettavissa Turingin koneilla, jolloin Turingin koneella voi tehdä kaiken sellaisen laskennan, mihin on olemassa algoritmi. Yhtäpitävyyden perusteella myös mikä tahansa Turingin koneella toteutettavissa oleva laskenta voidaan esittää algoritmina.

2.3 Vaativuusluokat P, NP ja co-NP

Vaativuusluokka on joukko funktioita, jotka voidaan laskea annetuissa resurssirajoissa. Rajoitettuja resursseja voivat olla esimerkiksi aika ja tila. Tässä tutkielmassa keskitytään annetun ajan rajallisuuteen. Algoritmi mielletään ajan käytön kannalta tehokkaaksi, jos sen suorittamiseen ei kulu tarpeettoman paljoa aikaa.

Algoritmin tehokkuudessa on etenkin suurilla syötteillä selkeä ero riippuen siitä, toimiiko se eksponentiaalisessa vai polynomisessa ajassa. Jos syöte on suuri, eksponentiaalisen aikavaativuuden algoritmi toimii huomattavasti hitaammin kuin polynomiaikainen algoritmi. Tämän vuoksi polynomiaikaisia algoritmeja voidaan yleisesti ottaen pitää tehokkaina. On tietenkin mahdollista, että jossain tapauksessa, esimerkiksi pelkästään pienillä syötteillä operoitaessa, eksponentiaalisenkin aikavaativuuden algoritmi on tehokas.

Seuraavaksi määritellään aikavaativuusluokka DTIME.

Määritelmä 2.8. Olkoon t funktio luonnollisilta luvuilta positiivisille reaaliluvuille. *Aikavaativuusluokka*, DTIME($t(n)$), on kaikkien $O(t(n))$-aikaisella Turingin koneella ratkeavien kielten kokoelma.

Luokkaan DTIME kuuluvat nimenomaan määritelmän 2.5 mukaisella deterministisellä yksinauhaisella Turingin koneella ratkeavat kielet. Määritellään seuraavaksi vaativuusluokka P.

Määritelmä 2.9. $\mathrm{P} = \bigcup_k \mathrm{DTIME}(n^k)$.

Luokka P on siis deterministisellä yksinauhaisella Turingin koneella polynomisessa ajassa ratkeavien kielten luokka. Luokkaan P kuuluvien kielten polynomisessa ajassa ratkaisevia algoritmeja sanotaan polynomiaikaisiksi algoritmeiksi. Kielen voidaan osoittaa kuuluvan luokkaan P esittämällä polynomiaikainen algoritmi, joka ratkaisee kyseisen kielen.

Tehokas algoritmi käyttää laskenta-askelia vain polynomisen määrän syötteen kokoon nähden. Esimerkiksi luonnollisella luvulla n tämä tarkoittaa siis polynomista määrää laskenta-askelia luvun n binääriesityksen pituuden, $\lceil \log n \rceil$, suhteen.

Määritellään seuraavaksi aikavaativuusluokka epädeterministisesti halutussa ajassa ratkeaville kielille.

Määritelmä 2.10.

$$\mathrm{NDTIME}(t(n)) = \{L : L \text{ on Turingin koneella epädeterministisesti } O(t(n))\text{-ajassa ratkeava kieli}\}.$$

Epädeterministisen polynomisen tapauksen vaativuusluokka on NP, joka määritellään seuraavaksi.

Määritelmä 2.11. $\mathrm{NP} = \bigcup_k \mathrm{NDTIME}(n^k)$.

Luokassa NP ovat kielet, jotka ratkeavat polynomisessa ajassa epädeterministisen Turingin koneen joillain valinnoilla. Tämä ei kuitenkaan takaa, että ratkaisu saataisiin aina polynomisessa ajassa, sillä epädeterministinen Turingin kone voi pahimmillaan joutua kokeilemaan kaikki vaihtoehtoiset laskennat saamalleen syötteelle. Vaihtoehtoja voi olla hyvinkin paljon riippuen epädeterminismin mukanaan tuomien valintamahdollisuuksien määrästä.

Luokka co-NP on epädeterministisesti polynomiaikaisesti Turingin koneella ratkeavien kielten komplementtien luokka. Siihen kuuluvat siis kielet, joiden komplementit ovat luokassa NP.

Luku 3

Alkuluvut

Tässä luvussa esitellään alkuluvun tunnistamisen lukuteoreettinen pohja. Osiossa 3.1 käsitellään jaollisuutta ja kongruenssia. Tämän jälkeen osiossa 3.2 binomikertoimet yhdistetään jaollisuuteen. Osiossa 3.3 perehdytään alkulukuihin. Osiossa 3.4 käydään läpi historiallisia tapoja tunnistaa alkuluku. Osiossa 3.5 tutustutaan alkulukujen tunnistamiseen vaativuusteorian laskennallisena ongelmana.

3.1 Jaollisuus ja kongruenssi

Tässä osiossa aiheina ovat jaollisuus ja kongruenssi. Määritellään aluksi, mitä tarkoittaa, kun luku jakaa toisen luvun.

Määritelmä 3.1. Kokonaisluku a *jakaa* kokonaisluvun b, jos $ax = b$ jollain kokonaisluvulla x. Tässä tapauksessa lukua a sanotaan luvun b *jakajaksi* ja lukua b luvun a *monikerraksi*. Kun luku a jakaa luvun b, merkitään $a|b$.

Kokonaislukua toisella jaettaessa jako ei mene aina tasan. Tätä tilannetta varten määritellään jakojäännös ja käytettävä merkintä. Jaon mennessä tasan jakojäännös on nolla.

Määritelmä 3.2. Olkoot $a, n \in \mathbb{Z}$ ja $d \in \mathbb{Z}_+$. *Jakojäännöstä* r jaettaessa luku n luvulla d merkitään

$$n \bmod d = r,$$

kun $n = qd + r$ ja $0 \leq r < d$, missä r ja q ovat yksiselitteisesti määritettäviä kokonaislukuja.

Lukuja voi yhdistää ominaisuus, että ne saavat saman jakojäännöksen jollain luvulla jaettaessa. Tällöin nämä saman jakojäännöksen saavat luvut ovat kongruentteja keskenään, kuten seuraavasta määritelmästä käy ilmi.

Määritelmä 3.3. Olkoon $m \geq 2$ annettu sekä a ja b mielivaltaisia kokonaislukuja. Luku a on *kongruentti* luvun b kanssa *modulo* m, jos $a \mod m = b \mod m$. Tämä merkitään $a \equiv b \pmod{m}$.

Määritellään seuraavaksi kahden kokonaisluvun suurin yhteinen tekijä ja pienin yhteinen jaettava.

Määritelmä 3.4. Kahden kokonaisluvun a ja b, joista molemmat eivät ole nollia, *suurin yhteinen tekijä* on suurin kokonaisluku, joka jakaa sekä luvun a että luvun b.

Määritelmä 3.5. Kahden kokonaisluvun a ja b *pienin yhteinen jaettava* on pienin positiivinen kokonaisluku, joka on jaollinen sekä luvulla a että luvulla b.

Lukujen a ja b suurin yhteinen tekijä merkitään (a, b) tai $\text{syt}(a, b)$. Kahden nollan suurin yhteinen tekijä määritellään olemaan nolla, eli $(0, 0) = 0$. Lukujen a ja b pienin yhteinen jaettava merkitään $[a, b]$ tai $\text{pyj}(a, b)$.

3.2 Binomikertoimet

Binomikertoimet saadaan kaavasta

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!},$$

joka kertoo n-alkioisen joukon k alkiota käsittävien osajoukkojen eli *kombinaatioiden* lukumäärän. Nolla alkiota sisältäviä osajoukkoja on yksi, jolloin binomikerroin saa arvon $\binom{n}{0} = 1$.

Seuraavana on tärkeä yhtäpitävyys binomikertoimille.

Lause 3.6. *Olkoon n ja k positiivisia kokonaislukuja siten, että $n \geq k$. Tällöin*

$$\binom{n}{k} + \binom{n}{k-1} = \binom{n+1}{k}.$$

Todistus.

$$\begin{aligned}\binom{n}{k} + \binom{n}{k-1} &= \frac{n!}{k!\,(n-k)!} + \frac{n!}{(k-1)!(n-k+1)!} \\ &= \frac{n!(n-k+1)}{k!(n-k+1)!} + \frac{n!k}{k!(n-k+1)!} \\ &= \frac{n!((n-k+1)+k)}{k!(n-k+1)!} \\ &= \frac{n!(n+1)}{k!(n-k+1)!} \\ &= \frac{(n+1)!}{k!(n-k+1)!} \\ &= \binom{n+1}{k}.\end{aligned}$$

□

Binomikerrointen yhteys polynomeihin käy ilmi seuraavasta lauseesta.

Lause 3.7 (binomilause)**.** *Kaikille luonnollisille luvuille n*

$$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^k y^{n-k}.$$

Todistus. Yhtälön vasemmasta puolesta saadaan

$$(x+y)^n = (x+y)(x+y)\dots(x+y),$$

jossa on n tekijää. Kun avataan sulut, saadaan summa monesta termistä, jotka ovat muotoa $x^k y^{n-k}$ jollain k. Termit saadaan valitsemalla x:t k suluista ja y:t lopuista suluista. Määrä tapoja, joilla k sulut voidaan valita, on $\binom{n}{k}$. Siten termin $x^k y^{n-k}$ kerroin on $\binom{n}{k}$. □

Seuraava lemma on hyödyllinen luvun 5 alkuluvun tunnistavan algoritmin kannalta.

Lemma 3.8. *Olkoon LCM(m) pienin yhteinen jaettava luvuille $1, \dots, m$. Kun $m \geq 7$, niin LCM(m) $\geq 2^m$.*

Todistus. Tarkastellaan integraalia

$$I(k,n) = \int_0^1 x^{k-1}(1-x)^{n-k}dx,$$

jossa $1 \leq k \leq n$. Hyödyntäen binomilausetta (lause 3.7) saadaan

$$\begin{aligned}
&(1-x)^{n-k} \\
=&(1+(-x))^{n-k} \\
=&\sum_{r=0}^{n-k} \binom{n-k}{r} \cdot 1^r \cdot (-x)^{(n-k)-r} \\
=&\sum_{r=0}^{n-k} \binom{n-k}{r} (-x)^r \\
=&\sum_{r=0}^{n-k} (-1)^r \binom{n-k}{r} x^r,
\end{aligned}$$

jossa $1 \leq k \leq n$.

Sijoitetaan saatu summalauseke integraaliin $I(k,n)$, jolloin saadaan

$$\begin{aligned}
I(k,n) &= \int_0^1 x^{k-1}(1-x)^{n-k}dx \\
&= \int_0^1 x^{k-1} \sum_{r=0}^{n-k} (-1)^r \binom{n-k}{r} x^r dx \\
&= \sum_{r=0}^{n-k} (-1)^r \binom{n-k}{r} \int_0^1 x^{k-1}x^r dx \\
&= \sum_{r=0}^{n-k} (-1)^r \binom{n-k}{r} \int_0^1 x^{k+r-1} dx \\
&= \sum_{r=0}^{n-k} (-1)^r \binom{n-k}{r} \frac{1}{k+r},
\end{aligned}$$

jossa $1 \leq k \leq n$. Siis $I(k,n) = \int_0^1 x^{k-1}(1-x)^{n-k}dx = \sum_{r=0}^{n-k}(-1)^r\binom{n-k}{r}\frac{1}{k+r}$. Summassa $0 \leq r \leq n-k$ eli $k \leq k+r \leq n$, joten nollaa suurempana ja korkeintaan luvun n suuruisena luku $k+r$ jakaa lukujen $1, \ldots, n$ pienimmän yhteisen jaettavan LCM(n). Tämän seurauksena LCM$(n)I(k,n) \in \mathbb{Z}$, kun $1 \leq k \leq n$.

Toisaalta iteroimalla $I(k,n) = \frac{(n-k)!}{n\cdot(n-1)\cdots k} = \frac{(n-k)!(k-1)!}{n!} = \frac{1}{k\binom{n}{k}}$, minkä seurauksena

$$k\binom{n}{k} | \text{LCM}(n) \quad \text{kaikilla} \quad 1 \leq k \leq n.$$

Tämän perusteella

$$n\binom{2n}{n} | \text{LCM}(2n)$$

ja

$$(2n+1)\binom{2n}{n} = (n+1)\binom{2n+1}{n+1} | \mathrm{LCM}(2n+1).$$

Nyt, koska $\mathrm{LCM}(2n) | \mathrm{LCM}(2n+1)$ ja $(n, 2n+1) = 1$, saadaan

$$n(2n+1)\binom{2n}{n} | \mathrm{LCM}(2n+1).$$

Tästä seuraa, että

$$\mathrm{LCM}(2n+1) \geq n(2n+1)\binom{2n}{n} \geq n \cdot 4^n \geq 2^{2n+2}, \text{ kun } n \geq 4.$$

Toiseksi viimeinen epäyhtälö pätee, koska $\binom{2n}{n}$ on suurin kerroin $(1+1)^{2n}$:n binomilaajennuksessa, jolloin $(1+1)^{2n} \leq \binom{2n}{n}(2n+1)$. Myös

$$\mathrm{LCM}(2n+2) \geq \mathrm{LCM}(2n+1) \geq 2^{2n+2}, \text{ kun } n \geq 4.$$

Saadaan siis $\mathrm{LCM}(2n+1) \geq 2^{2n+2} \geq 2^{2n+1}$ ja $\mathrm{LCM}(2n+2) \geq 2^{2n+2}$, kun $n \geq 4$. Näin ollaan osoitettu, että

$$\mathrm{LCM}(m) \geq 2^m, \text{ kun } m \geq 9.$$

Tarkastetaan vielä lukujen 7 ja 8 tapaukset. Kun $m = 7$, $\mathrm{LCM}(7) = 420 \geq 128 = 2^7$. Kun $m = 8$, $\mathrm{LCM}(8) = 840 \geq 256 = 2^8$. Näistä seuraa, että kun $m \geq 7$, niin $\mathrm{LCM}(m) \geq 2^m$.

□

3.3 Alkuluvut

Tässä osiossa käsitellään alkulukujen ja yhdistettyjen lukujen eroja sekä tutustutaan joihinkin alkulukuihin liittyviin tärkeisiin käsitteisiin ja tuloksiin. Määritellään ensiksi, mitkä luvut ovat alkulukuja ja mitkä eivät.

Määritelmä 3.9. Luku on *alkuluku*, jos se on lukua yksi suurempi positiivinen kokonaisluku, joka on jaollinen vain itsellään ja luvulla yksi.

Määritelmä 3.10. Luku on *yhdistetty luku*, jos se on lukua yksi suurempi positiivinen kokonaisluku, joka ei ole alkuluku.

Siis jokainen lukua yksi suurempi positiivinen kokonaisluku on joko alkuluku tai yhdistetty luku.

Lemma 3.11. *Olkoon $n \in \mathbb{N}$. Jos $n \geq 2$, n on jaollinen jollain alkuluvulla.*

Todistus. Jos n on alkuluku, se on jaollinen itsellään. Jos n ei ole alkuluku, se on jaollinen jollakin luvulla $1 < n_1 < n$. Luku n_1 on joko alkuluku tai sitten se on jaollinen luvulla $1 < n_2 < n_1$. Luvun n jakajista muodostuu jono $n > n_1 > n_2 > \ldots$. Jono ei voi olla ääretön, joten saavutetaan $n_t \geq 2$, joka on alkuluku. □

Seuraava lause kertoo alkulukujen määrästä.

Lause 3.12. *Alkulukuja on äärettömästi.*

Todistus. Olkoon $p_1, \ldots, p_s$ mielivaltainen äärellinen lista erillisistä alkuluvuista. Lemman 3.11 perusteella luku $n = p_1 \cdots p_s + 1$ on jaollinen jollain alkuluvulla. Jos n olisi jaollinen jollain alkuluvuista $p_1, \ldots, p_s$, niin luvuilla n ja $n - 1$ olisi yhteinen alkulukutekijä. Tällöin olisi oltava alkuluku, joka jakaa luvun $n - (n - 1) = 1$, mikä on mahdotonta. Luku n ei siis voi olla jaollinen millään alkuluvuista $p_1, \ldots, p_s$, joten se on jaollinen jollain muulla alkuluvulla. □

Määritellään seuraavaksi suhteelliset alkuluvut.

Määritelmä 3.13. Kokonaisluvut a ja b ovat *suhteellisia alkulukuja*, jos ne ovat keskenään jaottomia eli niiden suurin yhteinen tekijä on yksi.

Suhteellisiin alkulukuihin liittyy Eulerin ϕ-funktio, joka määritellään seuraavaksi.

Määritelmä 3.14. Olkoon n positiivinen kokonaisluku. *Eulerin ϕ-funktio* $\phi(n)$ kertoo niiden lukua n pienempien positiivisten kokonaislukujen määrän, jotka ovat suhteellisia alkulukuja luvun n kanssa.

Eulerin ϕ-funktio on multiplikatiivinen.

Korollaari 3.15. *Jos suhteellisilla alkuluvuilla n_1 ja n_2 pätee $n = n_1 n_2$, niin $\phi(n) = \phi(n_1) \cdot \phi(n_2)$.*

Todistus. Katso esimerkiksi [12], sivut 209-210. □

Jos n on alkuluku, $\phi(n) = n - 1$.

Lause 3.16 (Eulerin lause)**.** *Olkoon m positiivinen kokonaisluku ja a kokonaisluku, jolla $(a, m) = 1$. Tällöin $a^{\phi(m)} = 1 \pmod{m}$.*

Todistus. Katso esimerkiksi [12], sivut 203-204. □

Kaikille $n > 6$ pätee $\phi(n) \geq \sqrt{n}$, [2].

Propositio 3.17. *Jos kokonaisluvun $n \geq 1$ alkulukuhajotelma on $n = p_1^{k_1} \cdots p_r^{k_r}$ alkuluvuilla $p_1^{k_1}, \ldots, p_r^{k_r}$, niin*

$$\phi(n) = \prod_{1 \leq i \leq r} (p_i - 1) p_i^{k_i - 1} = n \cdot \prod_{1 \leq i \leq r} \left(1 - \frac{1}{p_i}\right).$$

Todistus. Katso esimerkiksi [12], sivu 208 ja sivut 210-211. □

Lause 3.18 (Fermat'n pieni lause)**.** *Kaikilla alkuluvuilla p ja positiivisilla kokonaisluvuilla a, jotka eivät ole jaollisia luvulla p, pätee*

$$a^{p-1} \equiv 1 \pmod{p}.$$

Todistus. Olkoot $a, 2a, \ldots, (p-1)a$ kokonaislukuja. Yksikään näistä luvuista ei ole jaollinen luvulla p, sillä jos $p | ja$, niin $p | j$, koska $p \nmid a$. Tässä päädytään ristiriitaan, sillä p ei voi jakaa lukua j, koska $1 \leq j \leq p-1$. Lisäksi yksikään kokonaisluvuista $a, 2a, \ldots, (p-1)a$ muodostuvista pareista ei ole keskenään kongruentteja modulo p. Tämä nähdään, kun oletetaan, että $ja \equiv ka \pmod{p}$, missä $1 \leq j \leq k \leq p-1$. Koska $(a, p) = 1$, saadaan $j \equiv k \pmod{p}$, mikä on ristiriita, sillä j ja k ovat lukua $p-1$ pienempiä positiivisia kokonaislukuja.

Kokonaisluvut $a, 2a, \ldots, (p-1)a$ muodostavat $(p-1)$-kokoisen kokonaislukujoukon, jossa yksikään luku ei ole kongruentti luvun nolla kanssa modulo p, joten tiedetään, että näiden lukujen $(a, 2a, \ldots, (p-1)a)$ pienimmät positiiviset jäännökset, jossain järjestyksessä, ovat kokonaisluvut $1, 2, \ldots, p-1$. Tämän seurauksena kokonaislukujen $a, 2a, \ldots, (p-1)a$ tulo on kongruentti lukujen $1, 2, \ldots, (p-1)$ tulon kanssa modulo p. Siis

$$a \cdot 2a \cdots (p-1)a \equiv 1 \cdot 2 \cdots (p-1) \pmod{p}.$$

Tästä saadaan

$$a^{p-1}(p-1)! \equiv (p-1)! \pmod{p}.$$

Koska $((p-1)!, p) = 1$, poistamalla $(p-1)!$ saadaan $a^{p-1} \equiv 1 \pmod{p}$. □

3.4 Alkuluvun tunnistaminen

Yksi olennainen aihe alkulukuihin liittyen on alkulukujen tunnistaminen. Tässä osiossa esitellään eräs varhainen menetelmä alkuluvun tunnistamiseen.

Menetelmä on Eratostheneen seula, joka kehitettiin antiikin Kreikassa. Menetelmässä ideana on käydä lukuja läpi valittuun lukuun n asti ja tutkia jokaisella alkuluvulla, mitkä lukua n pienemmistä luvuista se jakaa. Lukujoukon läpikäyntejä siis tulee niin monta kuin on lukua n pienempiä alkulukuja.

Käytännössä menetelmässä siis listataan luvut lukuväliltä $[2, n]$, jonka jälkeen valitaan ensimmäinen luku, jota ei ole yliviivattu tai todettu alkuluvuksi, eli luku kaksi, koska yhtäkään lukua ei ole aluksi käsitelty. Tämä luku on alkuluku, joten se merkitään alkuluvuksi. Viivataan yli kaikki muut yliviivaamattomat luvut, jotka ovat jaollisia kyseisellä luvulla. Kun on päästy listalla loppuun, siirrytään takaisin alkuun ja valitaan seuraava mahdollisimman pieni yliviivaamaton ja alkuluvuksi osoittamaton luku, jolla vaihe toistetaan. Lopuksi, kun kaikki luvut on joko viivattu yli tai todettu alkuluvuksi, nähdään, mitkä korkeintaan luvun n suuruiset luvut ovat alkulukuja. Menetelmä ei kuitenkaan ole tehokas suuria lukuja tutkittaessa.

3.5 PRIMES

PRIMES on kieli, joka koostuu kaikkien alkulukujen joukosta. Formaalisti tämä kieli voidaan määritellä seuraavasti:

$$\text{PRIMES} = \{\text{bin}(n) | n \geq 2 \text{ on alkuluku}\},$$

jossa $\text{bin}(n)$ tarkoittaa luvun n binääriesitystä.

Alkulukutestauksessa keskeinen kysymys on, onko annettu luku alkuluku. Tämä voidaan muotoilla päätösongelmaksi kysymällä, kuuluuko annettu luku alkulukuihin, eli kieleen PRIMES.

PRIMES kuuluu luokkaan P. Tästä on kerrottu tarkemmin luvussa 5. Ongelma kuuluu myös luokkiin NP ja co-NP.

Menetelmiä ongelman ratkaisemiseen on monia. Ennen Agrawalin, Kayalin ja Saxenan kehittämää algoritmia ongelman ratkaisemiseen pyrkineet algoritmit eivät ole olleet luokassa P. Ne eivät tuottaneet oikeaa vastausta täysin varmasti tai niihin liittyi jokin rajoittava ehto, jonka alla ne toimivat. Tällaisia ovat muun muassa satunnaisalgoritmit, joita ovat esimerkiksi Miller-Rabin algoritmi ja Solovay-Strassen algoritmi.

Luku 4

Äärelliset kunnat ja syklotomiset polynomit

Luvussa 5 esiteltävän ehdottoman deterministisen polynomiaikaisen alkuluvut tunnistavan algoritmin toimivuuden tutkimisessa tarvitaan jonkin verran algebraa. Tässä luvussa käydään läpi joitakin algebran peruskäsitteitä painottuen algebrallisiin rakenteisiin sekä perehdytään äärellisiin kuntiin ja syklotomisiin polynomeihin.

Luvun keskeinen päämäärä on tarkastella algebraa tutkielman kannalta oleellisilta osin. Aloitetaan määrittelemällä joitakin keskeisiä algebrallisia rakenteita osiossa 4.1. Tämän jälkeen osiossa 4.2 tutustutaan muun muassa juurikuntiin ja yksikköjuuriin sekä määritellään syklotomiset polynomit.

4.1 Ryhmät, renkaat ja kunnat

Ryhmä koostuu joukosta ja laskutoimituksesta. Ryhmä on suljettu laskutoimituksensa suhteen, siinä pätee liitännäisyys ja se sisältää laskutoimituksensa neutraalialkion ja käänteisalkiot jokaiselle alkiolleen.

Määritelmä 4.1. Joukko G varustettuna kaksipaikkaisella laskutoimituksella $\circ$ on *ryhmä*, jos seuraavat ehdot toteutuvat:

1. Kaikilla $g, h \in G$ pätee $g \circ h \in G$.
2. $g \circ (h \circ k) = (g \circ h) \circ k$ kaikilla $g, h, k \in G$.
3. On olemassa $e \in G$ siten, että $g \circ e = e \circ g = g$ kaikilla $g \in G$.
4. Kaikilla $g \in G$ on olemassa $h \in G$, jolla $g \circ h = h \circ g = e$.

Ryhmä G, jonka laskutoimitus on $\circ$ ja neutraalialkio e, voidaan merkitä myös $(G, \circ, e)$, jolloin laskutoimitus ja neutraalialkio nähdään merkinnästä.

Monoidi poikkeaa ryhmästä siten, että siinä ei ole alkioidensa käänteisalkioita.

Ryhmän *kertaluvulla* tarkoitetaan sen joukon mahtavuutta eli alkioiden lukumäärää. Ryhmälle G tämä merkitään $|G|$.

Määritellään seuraavaksi aliryhmä ja se, mitä tarkoittaa, jos aliryhmä on alkion virittämä.

Määritelmä 4.2. Olkoon $(G, \circ, e)$ ryhmä. Joukko $H \subseteq G$ on ryhmän G *aliryhmä*, jos H laskutoimituksen $\circ$ kanssa muodostaa ryhmän, jossa neutraalialkio on e.

Määritelmä 4.3. Kaikista ryhmän G alkion a potensseista koostuva ryhmän G aliryhmä on alkion a *virittämä* ja sitä merkitään $\langle a \rangle$.

Määritelmän mukaan siis $\langle a \rangle = \{a^i | i \in \mathbb{Z}\} = \{e, a, a^{-1}, a^2, a^{-2}, a^3, a^{-3}, \dots\}$. Tätä hyödynnetään myös seuraavassa syklisen ryhmän määritelmässä.

Määritelmä 4.4. Ryhmä $(G, \circ, e)$ on *syklinen*, jos ryhmässä on alkio a siten, että $G = \langle a \rangle$. Alkiota a kutsutaan ryhmän G *virittäjäalkioksi*.

Syklisissä ryhmissä myös alkioilla on kertaluvut.

Määritelmä 4.5. Olkoon $(G, \circ, e)$ ryhmä. Alkion $a \in G$ *kertaluku* $o_G(a)$ määritellään

$$o_G(a) = \begin{cases} |\langle a \rangle| & \text{, jos } \langle a \rangle \text{ on äärellinen} \\ \infty & \text{, muulloin.} \end{cases}$$

Yksi hyödyllinen tapaus alkion kertaluvusta on jollain alkuluvulla jaottoman luvun kertaluku modulo kyseinen alkuluku. Tälle on myös oma merkintänsä.

Määritelmä 4.6. Jos p on alkuluku ja n on luvulla p jaoton kokonaisluku, *luvun n kertaluku modulo p* määritellään $o_{\mathbb{Z}_p^*}(n \mod p)$ ja sitä merkitään $o_p(n)$.

Luvun a kertaluku modulo r on siis pienin luku k, jolla $a^k = 1 \pmod r$, kun $(a, r) = 1$, jossa $a \in \mathbb{Z}$ ja $r \in \mathbb{N}$. Kertaluvulle pätee, että $o_r(a) | \phi(r)$ millä tahansa a, jolla $(a, r) = 1$. Tämä nähdään, kun yhdistetään tiedot Fermat'n pienestä lauseesta (lause 3.18) ja Eulerin lauseesta (lause 3.16).

Kun H on ryhmän G aliryhmä, ryhmän G *vasemmat sivuluokat* modulo H määritellään $aH = \{ah : h \in H\}$. Merkintä G/H tarkoittaa näiden vasempien sivuluokkien joukkoa, $G/H = \{aH | a \in G\}$, ja sitä kutsutaan tekijäryhmäksi.

Rengas poikkeaa ryhmästä siinä, että renkaaseen liittyy kaksi laskutoimitusta. Rengas on kommutatiivinen ryhmä ensimmäisen laskutoimituksensa suhteen. Toiselle laskutoimitukselle pätee liitännäisyys, ja osittelulait ovat voimassa.

Määritelmä 4.7. Joukko R varustettuna yhteen- ja kertolaskulla, $+$ ja $\cdot$, on *rengas*, jos seuraavat ehdot pätevät:

1. Kaikille $a, b \in R$ pätee $a + b \in R$.
2. $a + (b + c) = (a + b) + c$ kaikilla $a, b, c \in R$.
3. On olemassa $0 \in R$ siten, että $a + 0 = 0 + a = a$ kaikilla $a \in R$.
4. Kaikilla $a \in R$ on olemassa $b \in R$, jolle $a + b = b + a = 0$.
5. $a + b = b + a$ kaikilla $a, b \in R$.
6. Kaikilla $a, b \in R$ pätee $a \cdot b \in R$.
7. $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ kaikilla $a, b, c \in R$.
8. On olemassa $1 \in R$ siten, että $a \cdot 1 = 1 \cdot a = a$ kaikilla $a \in R$.
9. $(a + b) \cdot c = a \cdot c + b \cdot c$ ja $c \cdot (a + b) = c \cdot a + c \cdot b$ kaikilla $a, b, c \in R$.

Mille tahansa renkaalle R sen *polynomirengas* on rengas $R[x]$, johon kuuluvat kaikki polynomit, joiden kertoimet ovat renkaassa R. Seuraavaksi määritellään renkaan erikoistapaus, kunta.

Määritelmä 4.8. Joukko R varustettuna yhteen- ja kertolaskulla, $+$ ja $\cdot$, on *kunta*, jos se on vaihdannainen rengas, jossa jokaisella nollasta poikkeavalla alkiolla on käänteisalkio.

Kunnan F multiplikatiivinen ryhmä F^* on kommutatiivinen ryhmä, joka koostuu joukosta $F \setminus \{0\}$ ja laskutoimituksesta $\cdot$ ja jolla on neutraalialkio 1.

Ryhmä $\mathbb{Z}_n^*$ koostuu joukosta $\{k | 1 \leq k \leq n-1, \text{syt}(k, n) = 1\}$ eli renkaan $\mathbb{Z}_n$ alkioista, joilla on käänteisalkio, ja kertolaskusta modulo n. Ryhmän koko voidaan ilmoittaa $|\mathbb{Z}_n^*| = \varphi(n)$.

Äärelliset kunnat ovat kuntia, joiden alkioiden määrä on äärellinen.

Lause 4.9. *Äärellisen kunnan kertaluku on alkulukupotenssi. Käänteisesti mille tahansa alkulukupotenssille on olemassa yksikäsitteinen äärellinen kunta, jonka kertaluku on kyseinen alkulukupotenssi.*

Todistus. Katso [4], sivu 273. □

Merkintä F_p tarkoittaa äärellistä p-alkioista kuntaa, jossa p on alkuluku. Jos p on alkuluku ja $h(X)$ on d-asteinen polynomi, joka on jaoton kunnassa F_p, niin sivuluokkien joukko $F_p[X]/(h(X))$ on äärellinen p^d-asteinen kunta. Merkintää $f(X) = g(X) \pmod{h(X), n}$ käytetään esittämään yhtälö $f(X) = g(X)$ renkaassa $\mathbb{Z}_n[X]/(h(X))$.

Esimerkki 4.10. Merkintä

$$(X+a)^n = X^n + a \pmod{X^r - 1, n}$$

tarkoittaa yhtälöä $(X+a)^n = X^n + a$ renkaassa $\mathbb{Z}_n[X]/(X^r - 1)$.

4.2 Syklotomiset polynomit

Luvussa 5 esiteltävän polynomiaikaisen algoritmin toimivuuden todistamisessa tarvitaan syklotomisia polynomeja. Tässä osiossa perehdytään niihin. Aloitetaan määrittelemällä renkaan ominaisuus karakteristika.

Määritelmä 4.11. Jos R mielivaltainen rengas ja on olemassa positiivinen kokonaisluku n siten, että $nr = 0$ kaikilla $r \in R$, niin pienin tällainen positiivinen luku n on renkaan R *karakteristika* ja renkaalla R sanotaan olevan positiivinen karakteristika n. Jos tällaista positiivista kokonaislukua n ei ole olemassa, renkaalla R sanotaan olevan karakteristika 0.

Kunnat ovat renkaita, joten myös kunnalla on karakteristika. Kunnan karakteristika vaikuttaa syklotomisiin polynomeihin.

Määritellään seuraavaksi juurikunta.

Määritelmä 4.12. Olkoon f n-asteinen polynomi, jolla $n > 0$, kunnan F suhteen. Polynomin f *juurikunta* K kunnan F suhteen on kunnan F laajennos, jolle pätee

a) $f(x) = c(x - \alpha_1)\dots(x - \alpha_n)$ renkaassa $K[x]$.

b) Yhdelläkään kunnan K aidolla kunnan F sisältävällä alikunnalla ei ole tätä ominaisuutta.

Polynomi jakautuu juurikunnassaan ensimmäisen asteen tekijöihin. Polynomin f juurikunta on siis pienin kaikki polynomin f juuret sisältävä kunta, joka määritellään polynomin f kertoimet sisältävän kunnan suhteen. Tämä tarkoittaa sitä, että juurikunta on polynomin kertoimet sisältävän kunnan laajennos, johon on lisätty kaikki polynomin f juuret.

Yhtenä tapauksena juurikunnasta määritellään seuraavaksi n:s syklotominen kunta ja siihen liittyen n:nnet yksikköjuuret.

Määritelmä 4.13. Olkoon n positiivinen kokonaisluku. Polynomin $x^n - 1$ juurikuntaa kunnan K suhteen kutsutaan *n:neksi syklotomiseksi kunnaksi* kunnan K suhteen ja merkitään $K^{(n)}$. Polynomin $x^n - 1$ juuria kunnassa $K^{(n)}$ kutsutaan *n:nsiksi yksikköjuuriksi* kunnan K suhteen ja näiden juurten joukkoa merkitään $E^{(n)}$.

Yksikköjuuresta voidaan käyttää myös nimitystä ykkösen juuri. Tätä nimentätapaa käyttäen n:s yksikköjuuri on ykkösen n:s juuri. Seuraavana on määritelmä n:nen yksikköjuuren erikoistapaukselle, primitiiviselle n:nelle yksikköjuurelle.

Määritelmä 4.14. Olkoot K kunta, jonka karakteristika on p, ja n positiivinen kokonaisluku, joka ei ole jaollinen luvulla p. Tällöin syklisen ryhmän $E^{(n)}$ virittäjää kutsutaan *primitiiviseksi n:neksi yksikköjuureksi* (tai *primitiiviseksi ykkösen n:neksi juureksi*) kunnan K suhteen.

Määritellään seuraavaksi n:s syklotominen polynomi.

Määritelmä 4.15. Olkoot K karakteristikan p kunta, n positiivinen kokonaisluku, joka ei ole jaollinen luvulla p, ja ζ primitiivinen n:s yksikköjuuri kunnan K suhteen. Tällöin polynomia

$$Q_n(x) = \prod_{\substack{s=1 \\ \text{syt}\,(s,n)=1}}^{n} (x - \zeta^s)$$

kutsutaan *n:neksi syklotomiseksi polynomiksi* kunnan K suhteen.

Seuraavat kaksi tulosta syklotomisten polynomien ominaisuuksiin liittyen ovat hyödyksi todistettaessa luvun 5 algoritmin toimivuutta.

Lause 4.16. *Olkoot K karakteristikan p kunta ja $Q_n(X)$ n:s syklotominen polynomi tässä kunnassa. Tällöin polynomi $Q_n(X)$ jakaa polynomin $X^n - 1$.*

Todistus. Katso [10], sivut 60-61. □

Syklotomiset polynomit jakautuvat tekijöihin.

Lause 4.17. *Jos $K = F_q$ ja $(q, n) = 1$, niin polynomi Q_n jakautuu jaottomiin d-asteisiin tekijöihin, jossa d on pienin positiivinen kokonaisluku siten, että $q^d = 1 \mod n$.*

Todistus. Katso [10], sivut 61-62. □

Luku 5

Polynomiaikainen algoritmi alkulukutestaukseen

Tässä luvussa esitetään Agrawalin, Kayalin ja Saxenan [1] löytämä ehdoton deterministinen polynomiaikainen algoritmi alkulukutestaukseen sekä sen toimivuuden ja aikavaativuuden todistus. Itse algoritmi ja idea sen taustalla on osiossa 5.1. Seuraavissa osioissa todistetaan algoritmin toimivuus. Osiossa 5.5 syvennytään algoritmin aikavaativuuteen.

5.1 Algoritmi

Tässä osiossa esitetään ensimmäinen ehdoton deterministinen polynomiaikainen algoritmi alkulukutestaukseen. Algoritmin keksiminen osoitti, että kysymys siitä, onko luku alkuluku, ratkeaa deterministisesti polynomisessa ajassa minkä tahansa luvun tapauksessa. Toisin sanoen siis sen, että ongelma PRIMES kuuluu luokkaan P.

Aloitetaan kuvaamalla algoritmin toiminta. Algoritmi saa syötteenä lukua yksi suuremman kokonaisluvun. Tästä luvusta algoritmi päättää, onko kysessä alkuluku vai yhdistetty luku. Todetessaan luvun alkuluvuksi algoritmi tulostaa PRIME, kun taas yhdistetyn luvun tapauksessa tulostetaan COMPOSITE. Algoritmissa on kuusi askelta, joista kolmessa se voi todeta luvun yhdistetyksi luvuksi ja kahdessa alkuluvuksi. Yksi askel suorittaa sopivan luvun etsinnän seuraavia varten. Algoritmi on esitetty kuvassa 5.1.

Syöte: kokonaisluku $n > 1$
1. Jos $n = a^b$, kun $a \in \mathbb{N}$ ja $b > 1$, tulosta COMPOSITE.
2. Etsi pienin r siten, että $o_r(n) > \log^2 n$.
3. Jos $1 < (a, n) < n$ jollain $a \leq r$, tulosta COMPOSITE.
4. Jos $n \leq r$, tulosta PRIME.
5. Kaikille $a \in [1, \lfloor \sqrt{\phi(r)} \log n \rfloor]$:
Jos $(X + a)^n \neq X^n + a \pmod{X^r - 1, n}$, tulosta COMPOSITE
6. Tulosta PRIME.

Kuva 5.1: Algoritmi.

Algoritmi perustuu seuraavaan Fermat'n pienen lauseen (lause 3.18) yleistykseen. Lemmaa voi hyödyntää alkuluvun tunnistamisessa, sillä se antaa kumpaankin suuntaan toimivan ehdon alkuluvuille.

Lemma 5.1. *Olkoon $a \in \mathbb{Z}$, $n \in \mathbb{N}$, $n \geq 2$ ja $(a, n) = 1$. Tällöin n on alkuluku, jos ja vain jos*

$$(X + a)^n = X^n + a \pmod{n}.$$

Todistus. Binomilauseen (lause 3.7) perusteella luvuilla $0 < i < n$ termin X^i kerroin lausekkeessa $((X + a)^n - (X^n + a))$ on $\binom{n}{i} a^{n-i}$.

Oletetaan aluksi, että n on alkuluku. Tällöin

$$\begin{aligned}\binom{n}{i} &= \frac{n!}{i!(n-i)!} \\ &= \frac{n(n-1)(n-2)\dots(n-i+1)}{i!} \\ &= n \cdot \frac{(n-1)(n-2)\dots(n-i+1)}{i!} \\ &= 0 \pmod{n}.\end{aligned}$$

Viimeinen yhtäsuuruus seuraa siitä, että $\binom{n}{i}$ on kokonaisluku ja että luvun n ollessa alkuluku $(n, i!) = 1$, joten luvun $\frac{(n-1)(n-2)\dots(n-i+1)}{i!}$ on oltava kokonaisluku. Luku n kerrottuna millä tahansa kokonaisluvulla modulo n on 0. Tämän perusteella kertoimet kaikille X^i-termeille, joilla $0 < i < n$, ovat nollia. Lisätään tähän Fermat'n pienestä lauseesta (lause 3.18) pienellä muokkauksella saatava tieto, että $a^n \equiv a \pmod{n}$, ja tieto, että $X^n - X^n = 0$, jolloin lausekkeessa $((X + a)^n - (X^n + a))$ myös X^0- ja X^n-termien tiedetään häviävän. Tämän seurauksena kaikki termit ovat nollia, eli $((X + a)^n - (X^n + a)) = 0$. Päädytään siis yhtälöön $(X + a)^n = X^n + a \pmod{n}$, kun n on alkuluku.

Oletetaan nyt, että n on yhdistetty luku. Tällöin luvulla n on alkulukutekijöitä. Tarkastellaan alkulukua q, joka on luvun n tekijä ja oletetaan,

että $q^k||n$. Siis $q^k|n$, mutta $q^{k+1} \nmid n$. Koska $q|n$ ja $(a,n)=1$, niin $(a,q)=1$, josta seuraa, että $(a^{n-q},q^k)=1$. Tällöin q^k ei jaa lukua $\binom{n}{q}$, mikä selviää tarkastelemalla vastaväitettä.

Oletetaan, että $q^k|\binom{n}{q}$. Tällöin jollain positiivisella kokonaisluvulla b pätee, että

$$\binom{n}{q} = \frac{n(n-1)(n-2)\dots(n-q+1)}{q!} = bq^k.$$

Tämä yhtäpitävyys voidaan esittää myös muodossa

$$n = \frac{b(q-1)!q^{k+1}}{(n-1)(n-2)\dots(n-q+1)},$$

josta tiedetään yhtälön oikealla puolella olevan luvun n olevan kokonaisluku. Olkoon $1 \leq j \leq q-1$. Oletetaan, että $q|(n-j)$, jonka seurauksena $n-j \equiv 0 \pmod{q}$. Luku q on luvun n tekijä, joten $n \equiv 0 \pmod{q}$, minkä seurauksena edellisen väitteen perusteella $j \equiv 0 \pmod{q}$. Tämä ei kuitenkaan ole totta, sillä $1 \leq j \leq q-1$. Siis $q \nmid (n-j)$ kaikilla $1 \leq j \leq q-1$. Tämän seurauksena

$$\frac{b(q-1)!}{(n-1)(n-2)\dots(n-q+1)}$$

on kokonaisluku, koska q on alkuluku. Yhtälössä pitäisi tästä johtuen päteä myös, että $\frac{n}{q^{k+1}}$ on kokonaisluku, josta seuraa, että $q^{k+1}|n$, mikä on ristiriita. Siis $q^k \nmid \binom{n}{q}$.

Siten X^q-termin kerroin ei ole nolla modulo n, sillä jos olisi, niin kyseinen kerroin olisi jaollinen luvulla n, eli $\binom{n}{q}a^{n-q} = cn$ jollain kokonaisluvulla c. Tästä seuraisi, että $q^k|\binom{n}{q}$, sillä $(q^k, a^{n-q})=1$ ja yhtälössä

$$\frac{\binom{n}{q}a^{n-q}}{q^k} = \frac{cn}{q^k}$$

oikea puoli olisi kokonaisluku, joten myös vasemman puolen olisi oltava.

Täten polynomi $((X+a)^n-(X^n+a))$ ei ole identtisesti nolla renkaassa $\mathbb{Z}_n$. Tästä seuraa, että yhtälö $(X+a)^n = X^n + a \pmod{n}$ ei päde, kun n ei ole alkuluku. Siis n on alkuluku, jos ja vain jos $(X+a)^n = X^n + a \pmod{n}$. □

Lemmasta 5.1 saadaan seuraavanlainen testi alkuluvuille. Annetulle syötteelle n valitaan a ja kokeillaan, toteutuuko kongruenssi

$$(X+a)^n = X^n + a \pmod{n}.$$

Tämä ei kuitenkaan ole tehokas tapa, koska yhtälön vasemmalla puolella voidaan joutua laskemaan suuruusluokkaa n oleva määrä kertoimia. Tapa

vähentää kerrointen määrää on laskea molemmat puolet yhtälöstä modulo muotoa $X^r - 1$ oleva polynomi, jossa r on valittu sopivan pieneksi. Toisin sanoen kokeillaan mikäli seuraava yhtälö pätee:

$$(X + a)^n = X^n + a \pmod{X^r - 1, n}.$$

Kuten aiemmin todettu, tällä merkinnällä tarkoitetaan, että tutkitaan yhtälöä $(X + a)^n = X^n + a$ renkaassa $\mathbb{Z}_n[X]/(X^r - 1)$.

Lemman 5.1 perusteella on selvää, että kaikki alkuluvut n toteuttavat yhtälön $(X+a)^n = X^n+a \pmod{X^r-1, n}$ kaikilla lukujen a ja r mahdollisilla arvoilla. Kuitenkin myös jotkin yhdistetyt luvut toteuttavat yhtälön joillakin lukujen a ja r arvoilla. Tästä huolimatta kuvattua testaustapaa voidaan käyttää melkein sellaisenaan. Jos sopivasti valittu luku r toteuttaa yhtälön useilla a, niin luvun n pitää olla alkulukupotenssi. Tämä käy ilmi myöhemmin algoritmin toimivuuden todistamisen yhteydessä. Lukujen a riittävää määrää ja sopivaa r rajoittaa $\log n$. Siten saadaan deterministinen polynomiaikainen algoritmi alkulukutestaukseen. Algoritmissa (kuva 5.1) riittäväksi määräksi lukuja a on todettu käydä läpi luvut $1 \leq a \leq \lfloor \sqrt{\phi(r)} \log n \rfloor$ ja sopiva r on pienin luku, jolla $o_r(n) > \log^2 n$.

5.2 Algoritmin oikeellisuus

Tässä ja kahdessa seuraavassa osiossa käydään läpi osiossa 5.1 esitetyn algoritmin (kuva 5.1) oikeellisuustodistus. Osion 5.4 lopussa on lause sille, että algoritmi palauttaa PRIME, jos ja vain jos n on alkuluku.

Tässä osiossa tarkastellaan algoritmin toimivuutta. Aloitetaan lemmalla, joka sisältää algoritmin toimivuuden ensimmäisen suunnan. Tämä sisältää osoituksen sille, että algoritmi tunnistaa kaikki alkuluvut alkuluvuiksi. Minkään alkuluvun kohdalla ei siis palauteta COMPOSITE, vaan mikä tahansa alkulukusyöte n tuottaa tuloksen PRIME.

Lemma 5.2. *Jos n on alkuluku, algoritmi (kuva 5.1) palauttaa PRIME.*

Todistus. Oletetaan, että n on alkuluku. Alkulukuna n ei ole jaollinen muilla kokonaisluvuilla kuin itsellään ja luvulla yksi. Luku n ei siis voi olla minkään luvun potenssi, eikä luvuilla a ja n voi olla suurinta yhteistä tekijää, jolla $1 < (a, n) < n$. Tällöin algoritmin askelet 1 ja 3 eivät voi palauttaa COMPOSITE. Lemman 5.1 perusteella myöskään vaiheen 5 silmukka ei voi palauttaa COMPOSITE, sillä kaikille alkuluvuille n pätee $(X+a)^n = X^n+a \pmod{X^r-1, n}$ kaikilla a ja r. Jäljellä ovat vaiheet 4 ja 6. Algoritmi tunnistaa näistä jommassa kummassa luvun n alkuluvuksi ja palauttaa siis PRIME. □

Nyt tiedetään, että algoritmi tunnistaa alkuluvut alkuluvuiksi. Tämän perusteella ei kuitenkaan vielä voida sanoa mitään syötteistä, jotka ovat yhdistettyjä lukuja. Saadessaan syötteeksi yhdistetyn luvun algoritmin pitäisi aina palautaa COMPOSITE ja ainoastaan alkuluvun tilanteessa PRIME. Siis jos algoritmi palauttaa PRIME, halutaan, että syöteluku n on alkuluku. Lähdetään tarkastelemaan tätä toista suuntaa algoritmin toimivuudesta.

Tarkastellaan tilannetta, että algoritmi palauttaa PRIME. Tämä voi tapahtua joko algoritmin vaiheessa neljä tai kuusi. Jos algoritmi palauttaa PRIME vaiheessa neljä, niin $n \leq r$. Algoritmi ei ole palauttanut COMPOSITE vaiheessa kolme, joten sen suurin yhteinen tekijä kaikkien itsestään poikkeavien lukujen $a \leq r$ kanssa on yksi. Koska $n \leq r$ ja $a \leq r$, niin kaikilla lukua n pienemmillä positiivisilla kokonaisluvuilla suurin yhteinen tekijä luvun n kanssa on yksi. Eli $(m, n) = 1$ kaikilla $m < n$. Tällöin luvun n täytyy olla alkuluku. Ainoa jäljellä oleva tilanne on siis se, että algoritmi palauttaa PRIME vaiheessa kuusi. Myöhemmissä tarkasteluissa oletetaan, että näin käy.

Algoritmin kaksi päävaihetta ovat sen toinen ja viides vaihe. Toisessa vaiheessa etsitään sopiva r ja viidennessä vaiheessa varmistetaan yhtälön $(X + a)^n = X^n + a \pmod{X^r - 1, n}$ toimivuus usealla luvulla a.

Seuraavan lemman avulla saadaan valittua sopiva r. Osoitetaan, että r saadaan rajattua tarpeeksi pieneksi.

Lemma 5.3. *On olemassa* $r \leq \max\left\{3, \lceil \log^5 n \rceil\right\}$ *siten, että* $o_r(n) > \log^2 n$.

Todistus. Kun $n = 2$, valinta $r = 3$ toteuttaa kaikki ehdot, sillä $o_3(2) = 2 > 1 = \log^2 2$.

Oletetaan, että $n > 2$. Tällöin $\lceil \log^5 n \rceil \geq \lceil \log^5 3 \rceil = 11 > 10$ ja voidaan soveltaa lemmaa 3.8. Olkoon $r_1, r_2, \ldots, r_t$ kaikki luvut alle annetun ylärajan siten, että joko $o_{r_i}(n) \leq \log^2 n$ tai r_i jakaa luvun n. Näiden lukujen kohdalla luvulta r vaadittu $o_r(n) > \log^2 n$ ei toteudu. Kaikki tällaiset luvut jakavat tulon

$$\begin{aligned}
n \cdot \prod_{i=1}^{\lfloor \log^2 n \rfloor} (n^i - 1) &< n \cdot \prod_{i=1}^{\lfloor \log^2 n \rfloor} n^i \\
&\leq n^{\left(\Pi_{i=1}^{\lfloor \log^2 n \rfloor} i\right)+1} \\
&\leq n^{\frac{\lfloor \log^2 n \rfloor \cdot \left(\lfloor \log^2 n \rfloor + 1\right)}{2}+1} \\
&< n^{\log^4 n} \\
&\leq \left(2^{\log n}\right)^{\log^4 n} \\
&\leq 2^{\log^5 n},
\end{aligned}$$

sillä jos $r_i|n$, niin r_i jakaa tulon ensimmäisen tekijän n, ja jos $o_{r_i}(n) \leq \log^2 n$, niin r_i jakaa jonkin myöhemmistä tulontekijöistä. Tämä johtuu siitä, että tulossa on tekijät $(n^1-1)(n^2-1)\dots(n^{\lfloor \log^2 n \rfloor}-1)$, joista jokin on 0 modulo r_i, kun $o_{r_i}(n) \leq \log^2 n$. Tämä seuraa kertaluvun ominaisuuksista. Nyt siis $o_{r_i}(n)$ on pienin positiivinen kokonaisluku k, jolla $n^k = 1 \pmod{r_i}$, eli $n^{o_{r_i}(n)}-1 = 0 \pmod{r_i}$. Tämä $n^{o_{r_i}(n)} - 1$ on jokin tulon $(n^1-1)(n^2-1)\dots(n^{\lfloor \log^2 n \rfloor}-1)$ tekijöistä, sillä $1 \leq o_{r_i}(n) \leq \log^2 n$.

Lemmasta 3.8 nähdään, että pienin yhteinen jaettava luvuille $1, \dots, \lceil \log^5 n \rceil$ on vähintään $2^{\lceil \log^5 n \rceil}$. Luvuista $1, \dots, \lceil \log^5 n \rceil$ kaikki eivät siis jaa tuloa

$$n \cdot \prod_{i=1}^{\lfloor \log^2 n \rfloor} (n^i - 1).$$

Tämän perusteella on olemassa luku $s \leq \lceil \log^5 n \rceil$ siten, että $s \notin \{r_1, r_2, \dots, r_t\}$. Luvut $r_1, r_2, \dots, r_t$ ovat kaikki luvut, joilla joko $o_{r_i}(n) \leq \log^2 n$ tai r_i jakaa luvun n. Luvulle s kumpikaan näistä ominaisuuksista ei siis päde. Luku s ei jaa lukua n ja jos kertaluku $o_s(n)$ voidaan määrittää, niin $o_s(n) > \log^2 n$.

Kertaluku $o_r(n)$ on määritettävävissä vain jos $(n, r) = 1$. Jos $(s, n) = 1$, niin $o_s(n) > \log^2 n$ ja siten haluttu luku on lyötynyt. Tarkastellaan tilanne, jossa $(s, n) > 1$, eikä s siten ole suoraan sopiva luvuksi r. Koska s ei jaa lukua n ja lukujen s ja n suurin yhteinen tekijä kuuluu kaikkien luvun n jakavien lukujen joukkoon, jonka seurauksena $(s, n) \in \{r_1, r_2, \dots, r_t\}$, niin jakamalla luvusta s lukujen s ja n suurin yhteinen tekijä pois saadaan $r = \frac{s}{(s,n)}$. Tällöin $(r, n) = 1$, jonka seurauksena $r \notin \{r_1, r_2, \dots, r_t\}$ ja siten luku r täyttää vaatimukset. Siis $o_r(n) > \log^2 n$. □

Edellisen lemman perusteella $o_r(n) > \log^2 n \geq \log^2 2 = 1$, joten on oltava olemassa luvun n alkulukujakaja p siten, että $o_r(p) > 1$. Tämä nähdään, jos oletetaan, että kaikilla luvun n alkulukutekijöillä p_i pätee $o_r(p_i) = 1$. Tällöin kaikille p_i pienin k, jolla $p_i^k = 1 \pmod{r}$, on $k = 1$. Luvulle n pätee, että $o_r(n)$ on pienin k, jolla $n^k = 1 \pmod{r}$. Luku n voidaan esittää tulona, jossa kaikki tekijät ovat luvun n alkulukutekijöitä, jolloin $n^k = (p_1 p_2 \dots p_j)^k = 1 \pmod{r}$, eli $n^k = p_1^k p_2^k \dots p_j^k = 1 \pmod{r}$. Kaikille p_i pätee $p_i^k = 1 \pmod{r}$ ja $1^c = 1$ riippumatta luvun c arvosta. Siten myös $o_r(n) = 1$, mikä on ristiriita. Siis on olemassa luvun n alkulukujakaja p siten, että $o_r(p) > 1$.

Tiedetään, että $p > r$ ja että $(n, r) = (p, r) = 1$, sillä muuten askeleessa kolme tai neljä olisi selvinnyt, onko luku n alkuluku vai ei. Jos $p \leq r$, se olisi lyötynyt vaiheessa kolme, sillä kyseisessä vaiheessa etsitään kaikki korkeintaan luvun r suuruiset luvut a, joilla $1 < (a, n) < n$, tai vaiheessa neljä tilanteen ollessa se, että $(p, n) = n$. Koska $(n, r) = (p, r) = 1$, niin $p, n \in \mathbb{Z}_r^*$. Kiinnitetään luvut p ja r oikeellisuustodistuksen loppuun saakka. Siis p on luvun

n jakava alkuluku ja luvulle r pätee, että $(r, n) = 1$, $r \leq \max\left\{3, \lceil \log^5 n \rceil\right\}$ ja $o_r(n) > \log^2 n$. Lisäksi merkitään $\ell = \lfloor \sqrt{\phi(r)} \log n \rfloor$.

Algoritmin viides vaihe tarkistaa, päteekö $(X+a)^n = X^n + a \pmod{X^r - 1, n}$ kaikilla luvuilla $1 \leq a \leq \lfloor \sqrt{\phi(r)} \log n \rfloor$. Koska oletuksena on, että algoritmi palauttaa PRIME vaiheessa kuusi, niin tarkastettavia yhtälöitä kertyy kaikki $\ell = \lfloor \sqrt{\phi(r)} \log n \rfloor$ kappaletta. Koska algoritmi palauttaa PRIME vaiheessa kuusi, eikä siis palauta COMPOSITE viidennessä vaiheessa, saadaan

$$(X + a)^n = X^n + a \pmod{X^r - 1, n}$$

kaikilla $0 \leq a \leq \ell$. Luku p jakaa luvun n, joten tästä seuraa

$$(X + a)^n = X^n + a \pmod{X^r - 1, p} \tag{5.4}$$

kun $0 \leq a \leq \ell$. Tiedetään p alkuluvuksi, joten lemman 5.1 perusteella saadaan

$$(X + a)^p = X^p + a \pmod{X^r - 1, p} \tag{5.5}$$

kun $0 \leq a \leq \ell$. Yhtälöiden 5.4 ja 5.5 ja tiedon $X^r - 1 | X^{\frac{nr}{p}} - 1$ perusteella saadaan

$$\begin{aligned}
&(X^{\frac{n}{p}} + a)^p \pmod{X^r - 1, p} \\
=&X^n + a \pmod{X^r - 1, p} \\
=&(X + a)^n \pmod{X^r - 1, p} \\
=&(X + a)^{p \cdot \frac{n}{p}} \pmod{X^r - 1, p}
\end{aligned}$$

kun $0 \leq a \leq \ell$. Tästä seuraa

$$(X + a)^{\frac{n}{p}} = X^{\frac{n}{p}} + a \pmod{X^r - 1, p} \tag{5.6}$$

kun $0 \leq a \leq \ell$. Täten kumpikin luvuista n ja $\frac{n}{p}$ käyttäytyy alkuluvun p tavoin ylläolevassa yhtälössä. Tätä ominaisuutta kutsutaan *introspektiivisuudeksi*. Seuraavassa osiossa perehdytään kyseiseen ominaisuuteen tarkemmin.

5.3 Introspektiivisuus

Tässä osiossa käsitellään introspektiivisuutta. Aloitetaan määrittelemällä introspektiivisuus.

Määritelmä 5.7. Olkoon $f(X)$ polynomi ja $m \in \mathbb{N}$. Sanotaan, että luku m on *introspektiivinen* polynomille $f(X)$, jos

$$[f(X)]^m = f(X^m) \pmod{X^r - 1, p}.$$

Yhtälöiden 5.6 ja 5.5 perusteella on selvää, että molemmat luvut, $\frac{n}{p}$ ja n, ovat introspektiivisiä polynomille $X + a$, kun $0 \leq a \leq \ell$.

Introspektiivisten lukujen joukko on suljettu kertolaskun suhteen.

Lemma 5.8. *Jos m ja m' ovat introspektiivisiä lukuja polynomille $f(X)$, niin myös $m \cdot m'$ on.*

Todistus. Oletetaan, että m ja m' ovat introspektiivisiä lukuja polynomille $f(X)$. Luvun m introspektiivisyydestä seuraa, että

$$[f(X)]^{m \cdot m'} = [f(X^m)]^{m'} \pmod{X^r - 1, p}.$$

Koska myös m' on introspektiivinen polynomille $f(X)$, saadaan

$$[f(X^m)]^{m'} = f(X^{m \cdot m'}) \pmod{X^{m \cdot r} - 1, p}.$$

Polynomi $X^r - 1$ jakaa polynomin

$$X^{m \cdot r} - 1 = (X^r - 1)(X^{r(m-1)} + X^{r(m-2)} + \cdots + X^r + 1),$$

joten

$$[f(X^m)]^{m'} = f(X^{m \cdot m'}) \pmod{X^r - 1, p}.$$

Näistä yhtälöistä saadaan

$$[f(X)]^{m \cdot m'} = f(X^{m \cdot m'}) \pmod{X^r - 1, p}.$$

Siis $m \cdot m'$ on introspektiivinen luku polynomille $f(X)$. □

Polynomien, joille luku m on introspektiivinen, joukko on suljettu kertolaskun suhteen.

Lemma 5.9. *Jos m on introspektiivinen polynomeille $f(X)$ ja $g(X)$, se on introspektiivinen myös polynomille $f(X) \cdot g(X)$.*

Todistus. Luku m on introspektiivinen polynomeille $f(X)$ ja $g(X)$, josta seuraa, että

$$\begin{aligned}[f(X) \cdot g(X)]^m &= [f(X)]^m \cdot [g(X)]^m \\ &= f(X^m) \cdot g(X^m) \pmod{X^r - 1, p}.\end{aligned}$$

Siis m on introspektiivinen myös polynomille $f(X) \cdot g(X)$. □

Nyt on tarvittavat tiedot introspektiivisuudesta, joten voidaan jatkaa algoritmin oikeellisuuden tarkastelun parissa.

5.4 Algoritmin oikeellisuuden todistaminen

Tässä osiossa todistetaan loppuun algoritmin toimivuus käyttäen apuna edellisen osion tietoja introspektiivisuudesta. Tähän mennessä tiedetään myös, että algoritmin tarkasteleman syöteluvun n alkulukujakajalle p pätee $o_r(p) > 1$. Luvulla r puolestaan pätee $r < p$, sen suurimmat yhteiset tekijät lukujen n ja p kanssa ovat $(r, n) = (p, r) = 1$, pätee rajoitus $r \leq \max\left\{3, \lceil \log^5 n \rceil\right\}$ ja $o_r(n) > \log^2(n)$. Käytetään myös merkintää $\ell = \lfloor \sqrt{\phi(r)} \log n \rfloor$.

Määritellään kaksi joukkoa, I ja P. Yhtälöiden 5.5 ja 5.6 perusteella luvut p ja $\frac{n}{p}$ ovat introspektiivisia polynomille $X + a$, joten lemmoista 5.8 ja 5.9 seuraa, että jokainen luku joukossa

$$I = \left\{ \left(\frac{n}{p}\right)^i \cdot p^j | i, j \geq 0 \right\}$$

on introspektiivinen kaikille polynomeille joukossa

$$P = \left\{ \prod_{a=0}^{\ell} (X + a)^{e_a} \, | e_a \geq 0 \right\}.$$

Näiden joukkojen pohjalta määritellään kaksi ryhmää, G ja $\mathcal{G}$.

Ryhmä G on joukko joukon I lukujen kaikista jakojäännöksistä modulo r, eli

$$G = \{i \pmod r | i \in I\}.$$

Kyseessä on renkaan $\mathbb{Z}_r$ yksiköiden multiplikatiivisen ryhmän $\mathbb{Z}_r^*$ aliryhmä, sillä G on ryhmä, jonka virittävät $p \pmod r$ ja $n \pmod r$, ja $(n, r) = (p, r) = 1$, jonka seurauksena p ja n ovat yksiköitä ryhmässä $\mathbb{Z}_r$. Olkoon ryhmän G alkioiden määrä $|G| = t$. Alkiot n ja p modulo r virittävät ryhmän G, sillä ryhmän G alkiot muodostuvat näiden kahden alkion potenssien tuloista. Koska $o_r(n) > \log^2 n$, niin $t > \log^2 n$, sillä jos alkiolla n potenssit $n^k \pmod r$, jossa $k \leq \log^2 n$, eivät tuota arvoa 1, niin näiden on kaikkien oltava eri alkioita ryhmässä G, minkä lisäksi ryhmässä on alkio 1.

Olkoon $Q_r(X)$ r:s syklotominen polynomi yli kunnan F_p. Lauseiden 4.16 ja 4.17 perusteella polynomi $Q_r(X)$ jakaa polynomin $X^r - 1$ ja jakautuu itse jaottomiin $o_r(p)$-asteisiin tekijöihin. Olkoon $h(X)$ yksi näistä jaottomista tekijöistä. Koska $o_r(p) > 1$, polynomin $h(X)$ aste on suurempi kuin yksi.

Ryhmä $\mathcal{G}$ on joukko kaikista joukon P polynomien jakojäännöksistä modulo $h(X)$ ja p, eli

$$\mathcal{G} = \{f \pmod{h(X), p} | f \in P\}.$$

Tämä ryhmä on selvästi alkioiden $X, X+1, X+2, \ldots, X+\ell$ virittämä kunnassa

$$F = F_p[X]/(h(X))$$

ja kunnan F multiplikatiivisen ryhmän aliryhmä.

Ryhmän $\mathcal{G}$ koko voidaan rajata. Seuraava lemma antaa alarajan ryhmän $\mathcal{G}$ koolle.

Lemma 5.10. $|\mathcal{G}| \geq \binom{t+\ell}{t-1}$.

Todistus. Huomataan ensiksi, että koska jaoton polynomi $h(X)$ on syklotomisen polynomin $Q_r(X)$ tekijä, X on primitiivinen r:s yksikköjuuri kunnassa F.

Seuraavaksi näytetään, että joukon P mitkä tahansa kaksi erillistä polynomia, joiden aste on pienempi kuin t, kuvautuvat eri alkioille ryhmässä $\mathcal{G}$. Olkoot $f(X)$ ja $g(X)$ tällaiset joukon P polynomit. Tehdään vastaoletus eli oletetaan, että $f(X) = g(X)$ kunnassa F. Olkoon $m \in I$. Myös $[f(X)]^m = [g(X)]^m$ kunnassa F. Koska m on introspektiivinen kummallekin polynomeista f ja g, ja $h(X)$ jakaa polynomin $X^r - 1$, saadaan

$$f(X^m) = g(X^m)$$

kunnassa F. Tästä seuraa, että X^m on polynomin $Q(Y) = f(Y) - g(Y)$ juuri kaikilla $m \in G$. Ryhmä G on multiplikatiivisen ryhmän $\mathbb{Z}_r^*$ aliryhmä, joten $(m, r) = 1$. Tämän seurauksena jokainen tällainen X^m on primitiivinen r:s yksikköjuuri. Siten polynomilla $Q(Y)$ on $|G| = t$ erillistä juurta kunnassa F. Kuitenkin polynomin $Q(Y)$ aste on pienempi kuin t johtuen polynomien f ja g valinnasta. Polynomilla on korkeintaan asteensa verran juuria, joten päädytään ristiriitaan, jonka perusteella $f(X) \neq g(X)$ kunnassa F. Nyt on osoitettu, että mitkä tahansa kaksi erillistä polynomia, joide aste on pienempi kuin t kuvautuvat eri alkioille ryhmässä $\mathcal{G}$.

Tiedetään pätevän $o_r(n) > \log^2 n$ ja $o_r(n) | \phi(r)$. Tämän avulla nähdään, että $o_r(n) < \phi(r) < r$, jonka seurauksena $r \log^2 n < r^2$ ja edelleen $\sqrt{r} \log n < r$. Tätä hyödyntäen huomataan, että $i \neq j$ äärellisessä kunnassa F_p, kun $1 \leq i \neq j \leq \ell$, koska $\ell = \lfloor \sqrt{\phi(r)} \log n \rfloor < \sqrt{r} \log n < r$ ja $p > r$. Siis alkiot $X, X+1, X+2, \ldots, X+\ell$ ovat kaikki erillisiä kunnassa F. Koska polynomin h aste on suurempi kuin yksi, pätee myös $X + a \neq 0$ kunnassa F kaikilla a, kun $0 \leq a \leq \ell$. Ryhmässä $\mathcal{G}$ on siis vähintään $\ell + 1$ erillistä ensimmäisen asteen polynomia. Muun asteiset ryhmän $\mathcal{G}$ polynomit saadaan näiden polynomien tuloista siten, että s-asteisen polynomin muodostukseen otetaan aina s kappaletta ensimmäisen asteen polynomeja, jotka saavat myös olla keskenään samoja. Tämän seurauksena polynomeja, joiden aste on pienempi kuin

t, on vähintään

$$\binom{\ell}{0} + \binom{\ell+1}{1} + \cdots + \binom{\ell+t-1}{t-1} = \sum_{s=0}^{t-1} \binom{\ell+s}{s}.$$

Käyttämällä toistuvasti lauseen 3.6 yhtäpitävyyttä

$$\binom{n}{k} + \binom{n}{k-1} = \binom{n+1}{k}$$

saadaan

$$\begin{aligned}
\binom{t+\ell}{t-1} &= \binom{\ell+t-1}{t-1} + \binom{\ell+t-1}{t-2} \\
&= \binom{\ell+t-1}{t-1} + \binom{\ell+t-2}{t-2} + \binom{\ell+t-2}{t-3} \\
&= \dots \\
&= \sum_{s=1}^{t-1} \binom{\ell+s}{s} + \binom{\ell+1}{0} \\
&= \sum_{s=1}^{t-1} \binom{\ell+s}{s} + 1 \\
&= \sum_{s=1}^{t-1} \binom{\ell+s}{s} + \binom{\ell}{0} \\
&= \sum_{s=0}^{t-1} \binom{\ell+s}{s},
\end{aligned}$$

jonka perusteella

$$\sum_{s=0}^{t-1} \binom{\ell+s}{s} \geq \binom{t+\ell}{t-1}.$$

Siten ryhmässä $\mathcal{G}$ on vähintään $\binom{t+\ell}{t-1}$ erillistä polynomia, joiden aste on pienempi kuin t.

□

Sen lisäksi, että ryhmän $\mathcal{G}$ koko voidaan rajoittaa alhaalta, kyseisen ryhmän alkiomäärälle on mahdollista määrittää myös yläraja. Seuraava lemma antaa ylärajan ryhmän $\mathcal{G}$ koolle. Kyseisessä lemmassa on kuitenkin rajoitus. Lemma toimii vain tilanteessa, jossa n ei ole p:n potenssi.

Lemma 5.11. *Jos n ei ole p:n potenssi, niin $|\mathcal{G}| \leq n^{\sqrt{t}}$.*

Todistus. Oletetaan, että n ei ole p:n potenssi. Tarkastellaan seuraavaa joukon I osajoukkoa:

$$\hat{I} = \left\{ \left(\frac{n}{p}\right)^i \cdot p^j | 0 \leq i, j \leq \lfloor \sqrt{t} \rfloor \right\}.$$

Luku n ei ole p:n potenssi, joten luvut $\left(\frac{n}{p}\right)$ ja p eivät voi olla sama luku tai toinen toisen potenssi. Tämän seurauksena tulosta $\left(\frac{n}{p}\right)^i \cdot p^j$ ei voi tulla sama tulos useilla eri arvoilla luvuille i ja j. Tällöin joukossa $\hat{I}$ on $\left(\lfloor \sqrt{t} \rfloor + 1\right)^2 > t$ erillistä alkiota, koska mahdollisia arvoja eksponentille i on $0, 1, \ldots, \lfloor \sqrt{t} \rfloor$ eli $\lfloor \sqrt{t} \rfloor + 1$ kappaletta, kuten myös arvoja eksponentille j. Siis $|\hat{I}| > t$. Koska $G = \{i \pmod r | i \in I\}$ ja $|G| = t$, vähintään kahden luvun joukossa $\hat{I}$ on oltava keskenään kongruentteja modulo r. Olkoot nämä luvut m_1 ja m_2, joille $m_1 > m_2$. Saadaan

$$\begin{aligned} & X^{m_1 - m_2} - 1 = 0 \pmod{X^r - 1} \\ \Leftrightarrow & \frac{X^{m_1}}{X^{m_2}} = 1 \pmod{X^r - 1} \\ \Rightarrow & X^{m_1} = X^{m_2} \pmod{X^r - 1}. \end{aligned}$$

Olkoon $f(X) \in P$. Tällöin luvut m_1 ja m_2 ovat introspektiivisia polynomille $f(X)$, sillä $m_1, m_2 \in \hat{I} \subset I$ ja kaikki luvut joukossa I ovat introspektiivisia kaikille polynomeille joukossa P. Saadaan

$$\begin{aligned} [f(X)]^{m_1} &= f(X^{m_1}) \pmod{X^r - 1, p} \\ &= f(X^{m_2}) \pmod{X^r - 1, p} \\ &= [f(X)]^{m_2} \pmod{X^r - 1, p}. \end{aligned}$$

Koska $F = F_p[X]/(h(X))$ ja $h(X) | X^r - 1$, niin tästä seuraa, että $[f(X)]^{m_1} = [f(X)]^{m_2}$ kunnassa F. Siten polynomi $f(X) \in \mathcal{G}$ on polynomin $Q'(Y) = Y^{m_1} - Y^{m_2}$ juuri kunnassa F. Koska $f(X)$ on mielivaltainen ryhmän $\mathcal{G}$ alkio, polynomilla $Q'(Y)$ on vähintään $|\mathcal{G}|$ erillistä juurta kunnassa F. Polynomille $Q'(Y) = Y^{m_1} - Y^{m_2}$ pätee $m_1 > m_2$ ja $m_1, m_2 \in \hat{I}$, joten

$$m_1 \leq \left(\frac{n}{p}\right)^{\lfloor \sqrt{t} \rfloor} \cdot p^{\lfloor \sqrt{t} \rfloor}.$$

Siten polynomin $Q'(Y)$ aste on

$$m_1 \leq \left(\frac{n}{p}\right)^{\lfloor \sqrt{t} \rfloor} \cdot p^{\lfloor \sqrt{t} \rfloor} \leq \left(\frac{n}{p} \cdot p\right)^{\lfloor \sqrt{t} \rfloor} \leq n^{\sqrt{t}}.$$

Polynomilla on korkeintaan asteensa verran juuria. Siis $|\mathcal{G}| \leq n^{\sqrt{t}}$.

□

Nyt voidaan näyttää toinen suunta algoritmin oikeellisuustodistuksesta. Tilanne on siis se, että algoritmi väittää syötteenä saamaansa lukua n alkuluvuksi eli palauttaa PRIME. Seuraava lemma osoittaa, että tällöin kyseessä todellakin on alkuluku.

Lemma 5.12. *Jos algoritmi (kuva 5.1) palauttaa PRIME, luku n on alkuluku.*

Todistus. Oletetaan, että algoritmi palauttaa PRIME. Lemmasta 5.10 seuraa, että kun $t = |G|$ ja $\ell = \lfloor \sqrt{\phi(r)} \log n \rfloor$:

$$|\mathcal{G}| \geq \binom{t+\ell}{t-1}.$$

Koska $t > \sqrt{t} \log n$, saadaan

$$|\mathcal{G}| \geq \binom{\ell + 1 + \lfloor \sqrt{t} \log n \rfloor}{\lfloor \sqrt{t} \log n \rfloor}.$$

Tästä edelleen, koska G on ryhmän $\mathbb{Z}_r^*$ aliryhmä, ryhmässä G on korkeintaan niin paljon alkioita kuin ryhmässä $\mathbb{Z}_r^*$, eli $\phi(r) \geq t$. Tällöin $\ell = \lfloor \sqrt{\phi(r)} \log n \rfloor \geq \lfloor \sqrt{t} \log n \rfloor$, joten

$$|\mathcal{G}| \geq \binom{2\lfloor \sqrt{t} \log n \rfloor + 1}{\lfloor \sqrt{t} \log n \rfloor}.$$

Koska $\lfloor \sqrt{t} \log n \rfloor > \lfloor \log^2 n \rfloor \geq 1$,

$$|\mathcal{G}| > 2^{\lfloor \sqrt{t} \log n \rfloor + 1}.$$

Saadaan

$$|\mathcal{G}| > 2^{\sqrt{t} \log n} \geq \left(2^{\log n}\right)^{\sqrt{t}} \geq n^{\sqrt{t}}.$$

Lemmasta 5.11 nähdään, että $|\mathcal{G}| \leq n^{\sqrt{t}}$, jos n ei ole luvun p potenssi. Edellisen perusteella tiedetään kuitenkin, että $|\mathcal{G}| > n^{\sqrt{t}}$, joten luvun n on oltava luvun p potenssi. Täten $n = p^k$ jollain $k > 0$. Algoritmin ensimmäisessä vaiheessa palautetaan COMPOSITE, jos $n = a^b$ joillain $a \in \mathbb{N}$ ja $b > 1$. Jos $k > 1$, niin n on tätä muotoa, joten algoritmi toteaa luvun yhdistetyksi luvuksi ja palauttaa COMPOSITE jo vaiheessa 1. Siten $n = p^1 = p$, eli n on alkuluku. Siis jos algoritmi väittää syötteenä annetun luvun n olevan alkuluku palauttamalla PRIME, luku n on alkuluku. □

Algoritmi toimii, sillä se tunnistaa jokaisen alkuluvun alkuluvuksi ja luvut, jotka eivät ole alkulukuja, yhdistetyiksi luvuiksi. Tämä esitetään seuraavassa lauseessa. Lause osoittaa algoritmin toimivan oikein.

Lause 5.13. *Algoritmi (kuva 5.1) palauttaa PRIME, jos ja vain jos n on alkuluku.*

Todistus. Lemman 5.2 perusteella luvun n ollessa alkuluku algoritmi palauttaa PRIME, ja lemman 5.12 perusteella PRIME palautetaan vain jos luku n on alkuluku. Yhdistämällä nämä tulokset saadaan, että algoritmi palauttaa PRIME, jos ja vain jos n on alkuluku. □

Algoritmi on nyt osoitettu toimivaksi. Seuraavaksi perehdytään sen aikavaativuuteen.

5.5 Algoritmin aikavaativuus

Tässä osiossa perehdytään osiossa 5.1 esitellyn algoritmin aikavaativuuteen. Aluksi todistetaan algoritmille aikavaativuus $O^{\sim}(\log^{21/2} n)$. Sen jälkeen esitetään lyhyesti eräs Agrawalin, Kayalin ja Saxenan artikkelissakin [1] mainittu parannus aikavaativuuteen.

Kahden m-bittisen luvun summa-, tulo- ja osamäärä-operaatiot voidaan kaikki suorittaa ajassa $O^{\sim}(m)$. Samoin nämä operaatiot kahdelle d-asteiselle polynomille, joiden kertoimet ovat korkeintaan m-bittisiä, voidaan suorittaa $O^{\sim}(d \cdot m)$ askeleeseen kuluvassa ajassa. Luvussa n bittien määrä on luvun n binaariesityksen pituus eli $\log n$. Siten korkeintaan luvun n suuruisten lukujen aritmeettiset operaatiot voidaan suorittaa ajassa $O^{\sim}(\log n)$ ja d-asteisille korkeintaan n-kertoimisille polynomeille ajassa $O^{\sim}(d \cdot \log n)$.

Potenssi a^n monoidissa voidaan laskea $O(\log n)$ kertolaskulla. Luonnollisesta luvusta n on mahdollista selvittää ajassa $O^{\sim}(\log^3 n)$, voidaanko se esittää kahden muun luonnollisen luvun a ja b avulla muodossa $n = a^b$. Lukujen n ja m suurin yhteinen tekijä voidaan laskea ajassa $O(\log^2 n)$. Algoritmit näiden toteutukseen löytyvät esimerkiksi Dietzfelbingerin kirjasta [5] sivuilta 69, 21 ja 28.

Nyt voidaan määrittää aikavaativuus kuvan 5.1 algoritmille.

Lause 5.14. *Algoritmin asymptoottinen aikavaativuus on $O^{\sim}(\log^{21/2} n)$.*

Todistus. Algoritmin ensimmäisessä vaiheessa selvitetään, onko $n = a^b$ millään $a, b \in \mathbb{N}$. Tämä vie ajan $O^{\sim}(\log^3 n)$.

Toisessa vaiheessa etsitään luku r, jolle $o_r(n) > \log^2 n$. Tämä voidaan tehdä kokeilemalla peräkkäisiä luvun r arvoja ja testaamalla josko $n^k \neq 1$

(mod r) kaikilla $k \leq \log^2 n$. Nimenomaiselle luvulle r, ja siten myös muille kokeiltaville luvun r arvoille, tämä sisältää enintään $O(\log^2 n)$ kertolaskua modulo r, jolloin kertolaskun tulontekijät ovat pienempiä kuin r, ja siten vie aikaa $O^{\sim}(\log^2 n \log r)$. Lemman 5.3 perusteella tiedetään, että riittää kokeilla $O(\log^5 n)$ eri vaihtoehtoa luvuksi r. Siten vaiheen 2 kokonaisaikavaativuus on $O^{\sim}(\log^7 n)$.

Kolmannessa vaiheessa lasketaan suurin yhteinen tekijä r lukuparille. Jokaisen suurimman yhteisen tekijän laskeminen voidaan toteuttaa ajassa $O(\log^2 n)$. Vaiheen aikavaativuus on siis $O(r \log^2 n)$. Lemman 5.3 perusteella $r \leq \max\{3, \lceil \log^5 n \rceil\}$, joten aikavaativuus tälle vaiheelle saadaan muotoon $O(\log^5 n \cdot \log^2 n) = O(\log^7 n)$.

Neljännen vaiheen aikavaativuus on $O(\log n)$.

Viidennessä vaiheessa tutkitaan yhtälön

$$(X + a)^n = X^n + a \pmod{X^r - 1, n}$$

pätevyys luvun a arvoilla $1 \leq a \leq \lfloor \sqrt{\phi(r)} \log n \rfloor$, joten täytyy tarkistaa $\lfloor \sqrt{\phi(r)} \log n \rfloor$ yhtälöä. Jokainen yhtälö vaatii $O(\log n)$ kertolaskua, sillä kerrottavia polynomeja lausekkeessa $(X + a)^n$ on n kappaletta ja ne ovat kaikki samoja, joten kerrottavien polynomien määrä puolittuu aina, kun yksi tulo on laskettu. Kertolaskut toteutetaan r-asteisilla polynomeilla, joiden kertoimet ovat kokoa $O(\log n)$, sillä kertoimet lasketaan yhtälössä modulo n. Siis jokainen yhtälö voidaan tarkistaa $O^{\sim}(r \log^2 n)$ askeleeseen kuluvassa ajassa. Tällöin vaiheen 5 aikavaativuus on

$$\begin{aligned}
&O^{\sim}(\sqrt{\phi(r)} \log n \cdot r \log^2 n) \\
&= O^{\sim}(r\sqrt{\phi(r)} \log^3 n) \\
&= O^{\sim}(r^{\frac{3}{2}} \log^3 n) \\
&= O^{\sim}(\log^{\frac{15}{2}} n \cdot \log^3 n) \\
&= O^{\sim}(\log^{21/2} n).
\end{aligned}$$

Vaiheen 5 aikavaativuus on suurin, joten se on myös koko algoritmin aikavaativuus. Siis algoritmin asymptoottinen aikavaativuus on $O^{\sim}(\log^{21/2} n)$. □

Lauseen 5.14 perusteella algoritmi toimii deterministisesti polynomisessa ajassa. Tämä siis osoittaa, että ongelma PRIMES kuuluu luokkaan P.

Algoritmin aikavaativuus saadaan pienemmäksi seuraavan lemman avulla luvun r arviota kehittämällä. Merkitään $P(m)$ luvun m suurinta alkulukujakajaa.

Lemma 5.15. *On olemassa vakiot $c > 0$ ja n_0 siten, että kaikille $x \geq n_0$ pätee*

$$|\left\{q : q \text{ on alkuluku, } q \leq x \text{ ja } P(q-1) > q^{\frac{2}{3}}\right\}| \geq c\frac{x}{\ln x}.$$

Todistus. Katso [6] ja [3]. □

Lause 5.16. *Algoritmin asymptoottinen aikavaativuus on $O^{\sim}(\log^{15/2} n)$.*

Todistus. Alkulukujen q, joilla $P(q-1) > q^{\frac{2}{3}}$, pakkautumisesta seuraa, että algoritmin toisessa vaiheessa voidaan löytää $r = O(\log^3 n)$, jolla $O_r(n) > \log^2 n$. Tämän seurauksena alogritmin aikavaativuudeksi saadaan $O^{\sim}(\log^{15/2} n)$. □

Aikavaativuus saadaan myös vielä alhaisemmaksi muokkaamalla algoritmia.

Kirjallisuutta

[1] Agrawal, Manindra; Kayal, Neeraj; Saxena, Nitin: PRIMES is in P, Annals of Mathematics, 160 (2004) 781-793.

[2] Arora, Sanjeev; Barak, Boaz: Computational Complexity: A Modern Approach, Cambridge University Press, 2009.

[3] Baker, R. C.; Harman, G.: The Brun-Titchmarsh Theorem on average. In: Berndt, B.C.; Diamond, H.G.; Hildebrand, A.J. (eds) Analytic Number Theory. Progress in Mathematics, 138 (1996) 39-103, Birkhäuser Boston.

[4] Cameron, Peter J.: Introduction to Algebra, 2nd edition, Oxford University Press, 2008.

[5] Dietzfelbinger, Martin: Primality Testing in Polynomial Time: From Randomized Algorithms to "PRIMES is in P", Springer, 2004.

[6] Fouvry, E.: Théorème de Brun-Titchmarsh; application au théorème de Fermat, Inventiones Mathematicae, 79 (1985) 383-407.

[7] Gathen, Joachim von zur; Gerhard, Jürgen: Modern Computer Algebra, 2nd edition, Cambridge University Press, Cambridge 1999.

[8] Häsä, Jokke; Rämö, Johanna: Johdatus abstraktiin algebraan, 2. painos, Gaudeamus, Helsinki, 2013.

[9] Leeuwen, Jan van (ed.): Handbook of Theoretical Computer Science, Volume A, MIT Press, 1990.

[10] Lidl, Rudolf; Niederreiter, Harald: Introduction to Finite Fields and their Applications, Cambridge University Press, 1986.

[11] Nair, M.: On Chebyshev-type inequalities for primes, The American Mathematical Monthly, 89 (1982) 126-129.

[12] Rosen, Kenneth H.: Elementary Number Theory and Its Applications, 3rd edition, Addison-Wesley, 1993.

[13] Sipser, Michael: Introduction to the Theory of Computation, 2nd edition, International edition, Thomson Course Technology, 2006.

[14] Smid, Michiel: Primality testing in polynomial time, 2003. Luettu 16.5.2018. Saatavilla http://people.scs.carleton.ca/∼ michiel/primes.pdf

[15] Wegener, Ingo: Complexity Theory: Exploring the Limits of Efficient Algorithms, Springer, 2005.